A Stochastic System Model for PageRank: Parameter Estimation and Adaptive Control

By

Cody E. Clifton

Submitted to the Department of Mathematics
and the Graduate Faculty of the University of Kansas
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Dr. Bozenna Pasik-Duncan, Chair

Dr. Tyrone E. Duncan

Dr. Joseph B. Evans

Dr. Zsolt Talata

Dr. Xuemin Tu

Date defended: April 20, 2015

The Dissertation Committee for Cody E. Clifton certifies
that this is the approved version of the following dissertation:

A Stochastic System Model for PageRank:
Parameter Estimation and Adaptive Control

Dr. Bozenna Pasik-Duncan, Chair

Date approved: April 20, 2015

Abstract

A key feature of modern web search engines is the ability to display relevant and reputable pages near the top of the list of query results. The PageRank algorithm provides one way of achieving such a useful hierarchical indexing by assigning a measure of relative importance, called the PageRank value, to each webpage. PageRank is motivated by the inherently hypertextual structure of the World Wide Web; specifically, the idea that pages with more incoming hyperlinks should be considered more popular and that popular pages should rank highly in search results, all other factors being equal.

We begin by overviewing the original PageRank algorithm and discussing subsequent developments in the mathematical theory of PageRank. We focus on important contributions to improving the quality of rankings via topic-dependent or "personalized" PageRank, as well as techniques for improving the efficiency of PageRank computation based on Monte Carlo methods, extrapolation and adaptive methods, and aggregation methods

We next present a model for PageRank whose dynamics are described by a controlled stochastic system that depends on an unknown parameter. The fact that the value of the parameter is unknown implies that the system is unknown. We establish strong consistency of a least squares estimator for the parameter. Furthermore, motivated by recent work on distributed randomized methods for PageRank computation, we show that the least squares estimator remains strongly consistent within a distributed framework.

Finally, we consider the problem of controlling the stochastic system model for PageRank. Under various cost criteria, we use the least squares estimates of the unknown parameter to iteratively construct an adaptive control policy whose performance, according to the long-run average cost, is equivalent to the optimal stationary control that would be used if we had knowledge of the true value of the parameter.

This research lays a foundation for future work in a number of areas, including testing the estimation and control procedures on real data or larger scale simulation models, considering more general parameter estimation methods such as weighted least squares, and introducing other types of control policies.

Acknowledgements

I would first like to extend my deepest gratitude to my advisor, Dr. Bozenna Pasik-Duncan, for imparting guidance, wisdom, and encouragement. Her infectious passion for mathematics is truly extraordinary.

I wish to thank Dr. Tyrone E. Duncan, Dr. Joseph B. Evans, Dr. Zsolt Talata, and Dr. Xuemin Tu for serving on my Doctoral Dissertation Committee; especially Dr. Duncan, whose expertise in stochastic control proved invaluable to my research.

I would like to recognize Mark Yannotta as being my original inspiration to study mathematics at an advanced level, and to thank him for encouraging me to pursue a graduate education at KU and for being my friend and mentor.

I would like to credit my longtime friends Theodore Lindsey and Forrest Koran with providing helpful feedback about the presentation of my research.

Finally, I wish to thank my family – including my mother, Lisa Clifton, my father, Dr. Kenneth E. Clifton, and my grandfather, Dr. H. Edward Clifton – for investing in my education, serving as my most cherished role models, fostering my widely-varied interests, and providing unconditional love and emotional support.

The research for this dissertation has been supported by the following grants of Dr. Tyrone E. Duncan and Dr. Bozenna Pasik-Duncan: FA9550-12-1-0384 (AFOSR), W911NF-10-1-0248 (ARO), DMS-0808138 and DMS-1108884 (NSF).

Contents

List of Figures

Chapter 1

Introduction

A key feature of modern web search engines is the ability to display relevant and reputable pages near the top of the list of query results. The PageRank algorithm, developed by Google co-founders Brin and Page in the late 1990s, provides one way of achieving such a useful hierarchical indexing by assigning a measure of relative importance, called the PageRank value, to each webpage.

1.1 Motivation

Submit a query to the Google search engine and in under a second it will return a list of results whose length is commonly between the thousands and the billions, as of the year 2015. It is technically possible to access all of this content sequentially, by clicking through page after page of results, and cleverly revising a search query may somewhat reduce the length of the list. However, the fact is that all but the most obscurely specific queries will return more information than can be practically inspected "by hand" in order to select the most desirable result(s), unless that information happens to be organized in some useful manner.

Fortunately, the information returned by Google is indeed organized. In fact, it is very often sorted so well that the first few search results contain the information we are searching for. How is this organization achieved? The answer lies partly in a method of indexing webpages, known as the PageRank algorithm, that has been used by Google since its inception.

Imagine a person surfing the web by randomly clicking on hyperlinks to travel between pages. The path of this "random surfer" can be viewed as a random walk on the vertices of the web graph and can be described mathematically as a trajectory of the Markov chain whose transition probabilities are determined by the hyperlink structure of the web. The likelihood that, in the long run, the surfer ends up on a specific page is entirely dependent on this hyperlink structure. In general, this long-term likelihood is sometimes called the stationary distribution of the Markov chain, but in the context of a random surfer traversing the web graph it is called PageRank.

In other words, the motivation for PageRank comes from the hypertextual structure of the World Wide Web; specifically, the idea that pages with more incoming links should be considered more popular and that popular pages should rank highly in search results, all other factors being equal. Nowadays, PageRank is but one of many metrics used by the Google search engine to index the web; other considerations include the order of appearance of search terms in content of a webpage, the recency or freshness of that content, and the geographical region from which a search originates [1].

In this dissertation, we introduce a model for PageRank whose dynamics are described by a stochastic linear system depending on an unknown parameter. We prove strong consistency of a sequence of least squares estimates of this parameter. Moreover, we employ these same least squares estimates in constructing adaptive control policies that behave optimally under a variety of cost criteria.

1.2 Organization

This document is comprised of five chapters in total, with the next three being organized as follows.

- Chapter 2 – We introduce the original PageRank algorithm and discuss subsequent developments in the mathematical theory of PageRank. Various examples are provided in order to reinforce key concepts.

- Chapter 3 – We present a model for PageRank whose dynamics are described by a controlled stochastic system that depends on an unknown parameter. The fact that the value of the parameter is unknown implies that the system is unknown. We establish strong consistency of a least squares estimator for the parameter. Furthermore, motivated by the recent work of Ishii and Tempo on distributed randomized methods for PageRank computation, we show that the least squares estimator remains strongly consistent within a distributed framework.

- Chapter 4 – We consider the problem of controlling the stochastic system model for PageRank. Under various cost criteria, we apply a control procedure that proceeds simultaneously with the parameter estimation described in the preceding chapter. In particular, we use the least squares parameter estimates to iteratively construct an adaptive control policy whose performance, according to the long-run average cost, is equivalent to the optimal stationary control that would be used if we had knowledge of the true value of the parameter.

Each of Chapters 2, 3, and 4 is concluded by a section containing numerically simulated examples. In Chapter 5, we summarize the contributions of this work and suggest some possible directions of future research.

Additionally, three appendices are included to complement the main document. In Appendix A, we present an introduction to discrete-time Markov chains on a finite state space,

which provides some prerequisite knowledge for the construction of PageRank in Chapter 2. Appendix B contains an introduction to discrete-time martingales, including a statement of the strong law of large numbers for martingales that is used repeatedly in Chapters 3 and 4. Finally, in Appendix C, we provide code for the MATLAB scripts and functions that are used to produce the numerical simulations that appear throughout the document.

1.3 Notation

The following notational conventions are assumed henceforth.

The letter C is reserved as a generic name for a positive, finite constant. To avoid confusion with subscripts denoting terms of a sequence, the components of a vector shall be represented with superscripts, e.g. $x = (x^{(1)}, \dots, x^{(n)})$. The closure of a set S, meaning the union of S and its boundary, is denoted by $\overline{S}$.

The prime, $'$, denotes transposition of a matrix or vector, while $\|\cdot\|$ is used to represent the Euclidean norm. The trace of a matrix M shall be written $\mathrm{tr}(M)$. The identity matrix is denoted simply by I, as its dimension will be clear from context. The phrases "M is non-negative definite" and "M is positive definite" are reserved for symmetric matrices and are denoted by $M \geq 0$ and $M > 0$, respectively. By $\tilde{M} \geq M$, we mean that $\tilde{M} - M \geq 0$, and the minimum of a set of matrices is taken with respect to this ordering.

Almost sure convergence, or convergence with probability one, is abbreviated as "a.s." To denote that the sequence $\{x_k, k = 0, 1, \dots\}$ converges almost surely to x as k approaches infinity, we will often write $x_k \xrightarrow{\text{a.s.}} x$ as $k \to \infty$. Other abbreviations that will appear repeatedly are "i.i.d." for "independent and identically distributed" and "SLLN" for the "strong law of large numbers."

Chapter 2

A Brief History of PageRank

The pioneering work on PageRank [9, 10, 45] by Google co-founders Brin and Page in the late 1990s paved the way not only for the emergence of today's most widely-used web search engine, but also for the investigation of a problem that has attracted researchers from many sub-disciplines of both mathematics and computer science. Important contributions have focused on improving the quality of rankings via topic-dependent or "personalized" PageRank [18, 28], as well improving the efficiency of PageRank computation using Monte Carlo methods [6, 49], extrapolation and adaptive methods [30, 32], and aggregation methods [11, 22, 26, 39, 37, 42, 52]. In this chapter, we present the original PageRank algorithm and discuss subsequent developments in the mathematical theory of PageRank.

The remainder of this chapter is organized as follows. In Section 2.1, we introduce the original PageRank algorithm developed by Brin and Page, while in Section 2.2 we explore existing mathematical results pertaining to PageRank, including generalizations of the algorithm and its use in applications beyond the indexing of webpages. Finally, in Section 2.3, we provide numerical simulations of PageRank computation.

2.1 Overview of PageRank

It is natural to describe a network of webpages and interconnecting hyperlinks from a graph-theoretic perspective. This allows us to discuss the behavior of Google's web crawlers (automated bots that navigate the web by following links from page to page) in the context of trajectories of a Markov chain. The stationary distribution of this Markov chain is what determines the PageRank value, or relative importance according to link structure, of each page within the web.

Consider a network of $n \geq 2$ webpages, indexed by the integers 1 to n, that is represented by the directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} := \{1, \ldots, n\}$ is the set of vertices corresponding to the page indices and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is the set of edges representing links between pages. In particular, $(i, j) \in \mathcal{E}$ if and only if page $i \in \mathcal{V}$ has an outgoing link to page $j \in \mathcal{V}$. Let $\mathcal{L}_i = \{j : (j, i) \in \mathcal{E}\}$ denote the set of indices corresponding to pages that link to page i, and let n_j be the total number of outgoing links of page j.

The *PageRank value*, or simply *PageRank*, of a page $i \in \mathcal{V}$ is a real number $X_*^{(i)} \in [0, 1]$ that represents the relative importance of i within the web, according to its link structure; in particular, $X_*^{(i)} > X_*^{(j)}$ implies that page i is more important than page j in terms of its incoming links. To achieve this sort of ranking, the value of each page is determined as a sum of contributions from all other pages in the web that link to it. Thus, we initially define $X_*^{(i)}$ by

$$X_*^{(i)} = \sum_{j \in \mathcal{L}_i} \frac{X_*^{(j)}}{n_j}.$$

A natural consequence of this definition is that pages with incoming links from important pages (i.e. pages with high PageRank) are themselves at least somewhat important.

2.1.1 Hyperlink matrix

Let $H = (h_{ij}) \in \mathbb{R}^{n \times n}$ be the *hyperlink matrix* defined for all $i, j \in \mathcal{V}$ by

$$h_{ij} := \begin{cases} \dfrac{1}{n_j}, & \text{if } j \in \mathcal{L}_i; \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

Observe that H is a nonnegative matrix, since all of its entries, h_{ij}, are nonnegative. Moreover, the columns of H are normalized by construction, provided that every page in the web has at least one outgoing hyperlink.

In the real web, pages with no outgoing links (e.g. multimedia content such as PDF, image, or video files) are commonplace and introduce columns of zeros in the hyperlink matrix H. Such so-called *dangling nodes* present a significant obstacle in the computation of PageRank that must be addressed. One straightforward way to circumvent this issue altogether is to give each dangling node an artificial outgoing link back to every page that links to it [24], as illustrated in Figure 2.1.

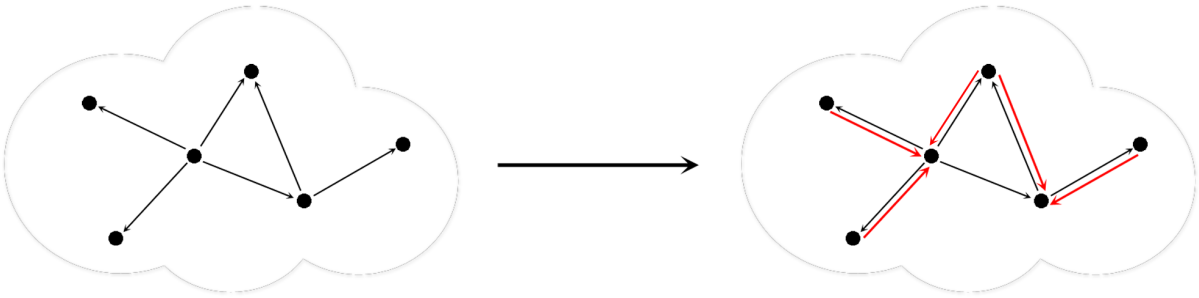


Figure 2.1: Introduction of artificial outgoing links from dangling nodes

The approach of using artificial links to eliminate dangling nodes in the web graph is justified by the concept of a "back button" (that is, a way of returning from any page to the immediately preceding page), which is a standard feature of modern web browsers. Various other approaches to dealing with dangling nodes have been considered and will be discussed in Section 2.2. In the meantime, we will simply assume that H is a nonnegative

column-stochastic matrix; that is, a nonnegative matrix with the property that $\sum_{i=1}^{n} h_{ij} = 1$ for $j = 1, \ldots, n$.

2.1.2 Random surfer model

Consider a web surfer who browses from page to page by clicking on links at random. Mathematically, the behavior of this random surfer is modeled by the Markov chain

$$X_{k+1} = HX_k, \ k = 0, 1, \ldots, \tag{2.2}$$

where $X_k \in [0,1]^n$ is a vector whose components, $X_k^{(i)}$, each represent the probability of the surfer being at a particular page, $i \in \mathcal{V}$, at time k. When the process $\{X_k, k = 0, 1, \ldots\}$ converges asymptotically, we find that its stationary distribution is precisely the *PageRank vector* $X_* = \left(X_*^{(1)}, \ldots, X_*^{(n)}\right) \in [0,1]^n$. Consequently, the computation of PageRank can be formulated as an eigenproblem:

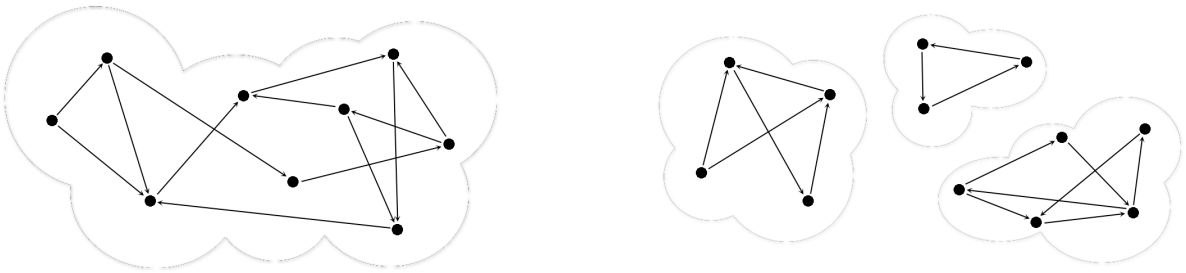
$$X_* = HX_*, \quad X_* \in [0,1]^n, \quad \sum_{i=1}^{n} X_*^{(i)} = 1,$$

where $H = (h_{ij})$ is the hyperlink matrix defined by (2.1).

2.1.3 Existence and uniqueness of PageRank

In general, for the eigenvector X_* corresponding to the eigenvalue 1 to exist uniquely, it is sufficient for the corresponding web graph to be strongly connected; that is, for all vertices $i, j \in \mathcal{V}$ there exists a sequence of edges connecting i to j. In other words, a strongly connected web is one in which any page can be reached from any other page simply by following the link structure. However, in the real world the web is known to *not* be strongly connected, so a modification of the basic algorithm is necessary in practice. Figures 2.2a and 2.2b contrast an "ideal" web graph that is strongly connected against a "real" web graph that consists of

multiple disconnected subgraphs.



(a) "Ideal" web is strongly connected (b) "Real" web is not strongly connected

Figure 2.2: "Ideal" web graph versus "real" web graph

In order to ensure the uniqueness of the eigenvalue 1, we let $t \in (0,1)$ be a parameter and let the modified hyperlink matrix $T \in \mathbb{R}^{n\times n}$ be defined by

$$T := (1-t)H + \frac{t}{n}\mathbf{1}\mathbf{1}', \tag{2.3}$$

where $\mathbf{1}\mathbf{1}' \in \mathbb{R}^{n\times n}$ is the matrix with all entries equal to 1. Henceforth, we will use the value of $t = 0.15$, which was proposed in [10] based on empirical observations. A more thorough discussion of this choice will be presented in Section 2.2.1.

By construction, T is a positive, stochastic matrix. Hence, the Perron-Frobenius theorem [44, p. 667] implies that 1 is a simple, dominant eigenvalue of T. Furthermore, the corresponding eigenspace is generated by a unique, positive, stochastic eigenvector. Therefore, we redefine the PageRank vector, X_*, in terms of T:

$$X_* = TX_*, \quad X_* \in [0,1]^n, \quad \sum_{i=1}^{n} X_*^{(i)} = 1.$$

2.1.4 Teleportation model

The modified definition of PageRank that uses T in place of H can still be interpreted in the context of a web surfer. Instead of simply navigating the web by following the link structure from page to page, the surfer may now occasionally become bored and *teleport* randomly

to a new page in the web that is not necessarily connected by hyperlinks (either directly or indirectly) to the current page. All n pages in the web have the same probability of being reached by such a random jump, and the probability that a jump occurs at any given time is represented by the parameter t. If a jump does not occur, then the surfer continues to navigate the web according to the transition probabilities given by H. Henceforth, t and T will be referred to as the *teleportation parameter* and *teleportation matrix*, respectively.

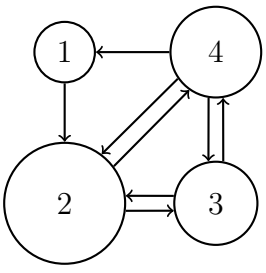
2.1.5 Power iteration for computing PageRank.

Due to the large dimension of H, computation of the eigenvector of T corresponding to the eigenvalue 1 is difficult. In practice, based on the power iteration method for determining the eigenvector corresponding to the largest eigenvalue of a matrix, X_* may be computed as the limiting distribution of the recursion

$$\begin{aligned} X_{k+1} &= TX_k \\ &= (1-t)HX_k + \frac{t}{n}\mathbf{1},\ k = 0, 1, \ldots, \end{aligned} \tag{2.4}$$

where $X_k \in [0,1]^n$ and X_0 is any probability vector. Observe that the recursion (2.4) requires only multiplication of the sparse matrix H rather than of the dense matrix T.

Example 2.1 (4-page Web)**.** Consider a small web of $n = 4$ pages with the following web graph and hyperlink matrix:



$$H = \begin{pmatrix} 0 & 0 & 0 & 1/3 \\ 1 & 0 & 1/2 & 1/3 \\ 0 & 1/2 & 0 & 1/3 \\ 0 & 1/2 & 1/2 & 0 \end{pmatrix}$$

In this case, the web graph is strongly connected, so it is not strictly necessary to employ the

teleportation model in order to achieve convergence of the power iteration to a probability vector that meaningfully differentiates the relative importance values of each of the four pages according to their link structure. However, we choose to do so in order to maintain consistency with the framework of the real World Wide Web. With the standard choice of $t = 0.15$ for the teleportation parameter, the PageRank vector for this small web is computed to be

$$X_* = (0.119, 0.331, 0.260, 0.289),$$

as reflected by the variable node sizing in the web graph diagram above.

We see that Page 1 has the smallest PageRank value, as expected since it has the smallest number of incoming links. Likewise, Page 2 has the largest value, since it has the largest number of incoming links. Pages 3 and 4 are similar in terms of their link structure, but Page 4 has marginally greater PageRank since it has one more outgoing link so it contributes less importance to Page 3 than Page 3 returns to it. In particular, Page 4 contributes only 1/3 of its own PageRank to Page 3, while Page 3 contributes 1/2 of its PageRank to Page 4.

In general, it follows from the Perron-Frobenius theorem that, for any probability vector $X_0 \in \mathbb{R}^n$, the power method recursion (2.4) yields convergence $X_k \to X_*$ as $k \to \infty$. Therefore, PageRank is guaranteed to be meaningful as a unique measure of the relative importance of each page in a web of arbitrary size, according to the hyperlink structure of that entire web. Some discussion of the convergence rate of the power iteration is provided in Section 2.2.1 below.

2.2 Existing Results in the Theory of PageRank

Google originally applied the PageRank algorithm to a web of 24 million pages in 1998. Now, seventeen years later, the size of the indexed web has grown thousand-fold to the order of tens of billions. As the web continues to grow, challenges arise in the computation of PageRank. Although these challenges are somewhat mitigated by improving computer technology, they

are also being addressed through ongoing academic research.

In this section, we present existing results pertaining to the PageRank algorithm, its convergence, variations, and applications to other areas. Special attention should be paid to Section 2.2.6 on the recent work of Ishii, Tempo, and Bai in the areas of web aggregation and distributed randomized PageRank computation. The latter will be revisited in Chapter 3, as we discuss the least squares estimation of an unknown parameter in a stochastic system model for PageRank.

2.2.1 Convergence and stability of the power iteration

Let $\lambda_1(T)$ and $\lambda_2(T)$ denote the eigenvalues of T with largest and second-largest magnitude, respectively. The asymptotic rate of convergence of the power iteration (2.4) is exponential and depends on the ratio $|\lambda_2(T)/\lambda_1(T)|$. Due to the fact that T is a positive, stochastic matrix, we have already established in Section 2.1.3 that $\lambda_1(T) = 1$ and $|\lambda_2(T)| < 1$. Using the structure of the link matrix, Haveliwala and Kamvar show in [19] that the modulus of the second eigenvalue satisfies the inequality

$$|\lambda_2(T)| \leq 1 - t,$$

where, again, $t \in (0, 1)$ is the teleportation parameter that represents the probability that a surfer becomes bored with following the link structure of the web and spontaneously jumps to a new page at random. Furthermore, if the transition matrix has at least two irreducible closed subsets (a fact that has been empirically verified for the web in practice), then we have equality

$$|\lambda_2(T)| = 1 - t.$$

It follows that the error level of the power method after k iterations is on the order of

$$\left|\frac{\lambda_2(T)}{\lambda_1(T)}\right|^k = (1-t)^k,$$

and therefore larger values of t imply faster convergence of the process $\{X_k, k = 0, 1, \dots\}$ to the true PageRank vector X_*.

In [19], Haveliwala and Kamvar remark that the greater the eigengap

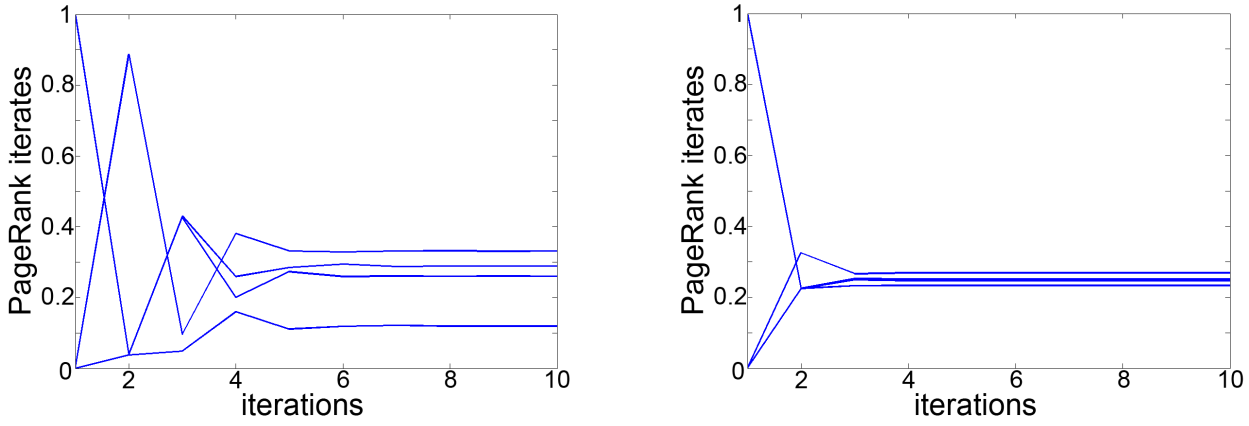
$$\begin{aligned} |\lambda_1(T)| - |\lambda_2(T)| &= 1 - (1 - t) \\ &= t, \end{aligned}$$

the more stable PageRank is with respect to perturbations of the link structure of the web. Moreover, they establish in [29] that the condition number of the PageRank problem is

$$\kappa(T) = \frac{2 - t}{t},$$

which further implies that a larger value of t corresponds to greater stability.

We see that there are multiple reasons for choosing a large value of the parameter $t \in (0, 1)$. However, there is one very important reason why t cannot be too large. Indeed, large values of t imply less dependence on the true structure of the web that is encoded in the original hyperlink matrix H, resulting in smaller differences between the PageRank values of distinct pages. This tradeoff is illustrated by Figure 2.3, in which the small web of Example 2.1 is revisited.



(a) $t = 0.15$ (b) $t = 0.9$

Figure 2.3: Power iteration with $t = 0.15$ versus $t = 0.9$ for a four-page web

From this example, we see that the parameter value $t = 0.9$ yields slightly more rapidly convergence in comparison with $t = 0.15$, but also causes the PageRank values of the four pages to be nearly indistinguishable from one another.

Thus, the choice of $t = 0.15$ is seen as a reasonable compromise; it provides an acceptable convergence rate and degree of stability according to early experiments by Brin and Page [10, 45], while maintaining the practicality of PageRank as a metric for distinctly ordering webpages by their importance according to hyperlink structure. With the choice of $t = 0.15$, the error level of the power method after 50 and 100 iterations is at most $0.85^{50} \approx 2.95 \times 10^{-4}$ and $0.85^{100} \approx 8.75 \times 10^{-8}$, respectively.

2.2.2 Improving rankings with personalized PageRank

In their original paper [10], Brin and Page suggest the potential of "personalizing" PageRank in order to improve the accuracy of search results. Whereas the original teleportation model assumes that teleportations will occur uniformly across the web, the idea behind personalized PageRank is to weight the possible teleportation destinations according to some useful criteria. This is accomplished by replacing the uniform probability vector $\frac{\mathbf{1}}{n}$ in (2.4) by a nonuniform probability vector, $\boldsymbol{p}$, called the "personalization vector." In [45], Page et al. explore the possibility of placing all the weight on a single page; that is, letting $\boldsymbol{p}_i = 1$ for some $i \in \mathcal{V}$ and $\boldsymbol{p}_j = 0$ for all $j \neq i$. Naturally, in such a case, the single page, i, that is always reached through teleportation has the highest PageRank, while its directly-connected neighbors in the web graph also rank highly.

Haveliwala contributed to the idea of PageRank personalization by introducing *topic-sensitive PageRank* [18]. This involves the usual pre-computation of importance scores, but with multiple scores, corresponding to different topics, being pre-computed for each page. This leads to more accurate search results, because the page ranking is tailored to the keywords in a given query. The actual personalization is carried out as described in [45], where the personalization vector in the PageRank algorithm is modified depending on

the topic to adjust the transition probabilities that relate to the teleportation procedure. By appropriately selecting the personalization vector, the rank vector can be made to prefer certain categories of pages. An experiment described in the paper, using 16 preselected topics for topic-sensitive personalization, demonstrates the improved accuracy of this method.

While using 16 different PageRank vectors, instead of one, may improve search accuracy, it comes at the cost of increased computational requirements. Since the computational gains realized by improving computer hardware are offset in the context of the PageRank problem by the ever-growing size of the web, the next step in practically implementing personalized rankings was to improve the efficiency of PageRank computation by developing more sophisticated theoretical approaches. From 2003 to 2004, Haveliwala collaborated with Kamvar to produce several papers to this effect.

In [31], the so-called BlockRank algorithm is introduced. It is the first method to exploit the inherent block structure of the web to speed up PageRank computation. The algorithm roughly consists of dividing the web up into blocks by domain, computing a local PageRank vector for each block, estimating the relative importance or *BlockRank* of each block, weighting each of the local PageRank vectors according to their corresponding BlockRank, and concatenating them to form a global PageRank vector that is then used as the initial approximation in the original PageRank algorithm. This method capitalizes on the fact that the local PageRank vectors for many blocks will converge in only a few iterations. Moreover, all local computations can be carried out in a parallel or distributed fashion.

The BlockRank algorithm works well in conjunction with the notion of personalized PageRank. Assuming that the personalization of teleportation probabilities is carried out at a block level (i.e. instead of being able to teleport according to a distribution over all pages in the web, a random surfer may teleport according to a distribution over the blocks; teleportation from a block to a page within it can be accounted for subsequently at minimal cost), the local PageRank vectors will not change for different personalizations, and neither does the block matrix that is used in computing the BlockRank weighting vector.

This eliminates much of the redundant computation that takes place in computing multiple personalized PageRank vectors using the original algorithm.

In addition to the topic-sensitive PageRank and BlockRank methods, a third approach to personalized PageRank was presented by Jeh and Widom in 2003 [28]. This Modular PageRank method restricts the choice of teleportation transitions in order to prefer a certain category of highly-ranked pages over arbitrary pages. For example, these preferred pages could be the set of bookmarks on a per-user basis, in which case the pages most closely related to a particular user's bookmarks would receive a higher ranking in their personalized search query results. Although this method provides a high degree of customizability, it has the drawback of requiring client-side computation in order to achieve near real-time updating of the set of preferred pages.

In [20], Haveliwala, Kamvar, and Jeh briefly compare their three approaches to personalizing PageRank. No conclusion is given as to the superiority of one approach over the others; each has its advantages in terms of the tradeoff between computational requirements and the granularity of personalization achieved. However, their ideas collectively proved lucrative. In 2003, following in the footsteps of Brin and Page, the three co-founded the company Kaltix in the interest of commercializing personalized web search technology. Kaltix was acquired by Google within a few months of its inception.

2.2.3 Other techniques for improving the accuracy of rankings

In 2002, Richardson and Domingos proposed a variation of PageRank, called QD-PageRank, that uses a directed (as opposed to random) surfer model in which the behavior of the surfer is affected by *query-dependent* probabilities [48]. Rather than following the hyperlink structure of the web at random, the surfer chooses links that lead to pages whose content is deemed relevant to a particular query according to some other criterion such as the frequency of appearance of keywords.

In 2004, Baeza-Yates and Davis suggested a modification of the original PageRank algo-

rithm that exploits some HTML features surrounding links in a webpage [7]. Their Weighted Links Rank, or WLRank algorithm, is characterized by three attributes: the tag in which the link appears (with more weight being given to links found within tags that convey emphasis, such as <h1>, , etc.), the length of the anchor text of the link (with more weight being given to links with detailed anchor text, and less being given to links described simply as *home* or *here*), and the relative position of the link on the page (with more weight being given to pages that appear at the beginning of the page than at the end, in the HTML source code view rather than in the browser view).

2.2.4 Extrapolation and adaptive methods

In [32], Kamvar, Haveliwala, Manning, and Golub present extrapolation methods for accelerating PageRank computation. The Aitken Extrapolation, based on an earlier method for accelerating linearly convergent sequences, assumes that the iterate $X(k-2)$ can be expressed as a linear combination of the first two eigenvectors of the transition matrix. Then, using the iterates $X(k-2)$, $X(k-1)$, and $X(k)$, the principal eigenvector (i.e. the PageRank vector) can be solved for in closed form. The Quadratic Extrapolation employs the same principle, but assumes that $X(k-3)$ can be written as a linear combination of the first three eigenvectors. Both of these methods are empirically demonstrated to work best in conjunction with the standard power method (specifically, when they are periodically applied between iterations of the power method) and the Quadratic Extrapolation performs the best in terms of time-savings due to increased convergence speed.

The final PageRank-related research contribution of Kamvar and Haveliwala came in 2004, when they presented an adaptive method for computing PageRank [30]. As with the BlockRank algorithm, this technique exploits the idea that the PageRank values for many individual pages converge rapidly, while for a few pages the values take much longer to converge. It proceeds by periodically re-partitioning the transition matrix and approximate PageRank vector to separate the pages whose PageRank values have converged from those

whose values have not. This eliminates much of the unnecessary re-computation of converged values that is inherent in the original algorithm. The Adaptive PageRank algorithm does not necessarily reduce the number of iterations required for convergence, since some pages may still take a large number of iterations to converge, but it does lower the average iteration cost.

2.2.5 Aggregation methods

Several aggregation methods have been applied to the PageRank problem. These all share the common goal of temporarily grouping certain pages or categories of pages together in order to reduce the size of the original hyperlink matrix, thereby mitigating the computational complexity of updating PageRank across a large web.

In [41], Lee, Golub, and Zenios present a two-stage algorithm for speeding up PageRank computation. The first stage involves lumping all dangling nodes into a single node and computing the PageRank vector for the resulting web graph. The second stage involves returning to the original web graph, weighting each of the non-dangling nodes using the values from the first stage, aggregating them into a single node, and then computing the PageRank vector for the resulting web graph. Concatenating the results of the two stages leads to an approximation of the complete PageRank vector. The overall computation time is observed to be proportional to the number of non-dangling nodes and the number of nonzero elements in the lumped transition matrix relative to the number of nonzero elements in the original matrix.

Multistage generalizations are also discussed. In particular, two examples are given. The first is a three-stage algorithm where the nodes are separated into three types: dangling, non-dangling, and weakly-dangling (i.e. pointing to a dangling node). The second is a variation of earlier personalization techniques, where customization with respect to classes of pages is enabled by using multiple personalization vectors in a multistage algorithm. This work is revisited in [42], where updated numerical experiments are provided along with a brief

discussion of embedding other acceleration methods, such as those of Kamvar et al. [32] and Arasu, Novak, Tomkins, and Tomlin [2], into the two-stage algorithm for additional speedup. In [22], Ipsen and Selee use the lumping technique of Lee et al. to develop a related algorithm that is more general because it allows the dangling node and personalization vectors to be different.

A reordering of the linear system formulation of the PageRank problem is presented by Langville and Meyer in [37] as a way to reduce computational complexity. This method uses strategic information about the impact of dangling nodes on the sparsity of the link matrix, and is equivalent to the two-stage Markov formulation approach of Lee et. al. [41]. Based on this reordering of the transition matrix, a reordered PageRank algorithm is introduced that provides further efficiency boosts by recursively locating zero rows in the original link matrix. The reordered algorithm has data-set-dependent speedup, but is theoretically shown to exhibit convergence that is no worse than the original algorithm.

In [34], Langville and Meyer introduce three new algorithms for updating PageRank. While the first two are observed to be computationally impractical, the third method, based on iterative aggregation-disaggregation (IAD) is not only promising from a computational standpoint but is also the first algorithm to address node updates (in addition to link updates) in an online manner.

In [36], Langville and Meyer use a 2-block IAD method is used to speed up PageRank computation. The state space is partitioned (and possibly reordered) so that the first block contains all newly added states from some update procedure along with some preexisting states, while the second block contains the remaining preexisting states. The states in the first block are left unaggregated, while those in the second block are aggregated into a single "super-state," the idea being that the PageRank values of those in the second block will not be greatly affected by the addition of the new pages accounted for in the first block. This technique significantly reduces the size of the matrix used for updating PageRank. In [39], the 2-block IAD method is revised in a more general framework of nearly completely

decomposable Markov chains. The application to PageRank updating is described in detail, and mention is made of the potential for combining this IAD method with the extrapolation method of Kamvar et al. [32].

In [52], Zhu, Ye, and Li introduce a distributed PageRank computation (DPC) algorithm, based on IAD methods. The basic idea of the DPC algorithm is to group pages that share certain properties, compute the local PageRank vector for each aggregated group, and then update the global PageRank vector through low volume communication with a coordinator. The DPC algorithm consists of roughly three steps. The first is a local step in which a local transition matrix is constructed for each "node" (that is, some block or cluster of pages, grouped according to reasonable criteria) and the local PageRank is computed for pages in each node. Next, a global step occurs, in which the PageRank distribution across all nodes is computed, based on the local PageRank approximations from the first step. Then, there is another local step, in which the local PageRank is re-computed at each node, based on the coarse level distribution from the global step. Ultimately, normalization yields the updated PageRank distribution approximation for all pages, at which point the algorithm may be reiterated or terminated.

Some advantages of the DPC are that it can be combined with existing acceleration methods (e.g. the extrapolation method of Kamvar et al. [32]), the matrices involved in the computations are small enough to fit into the main memory of a computer, and the local PageRank vectors for many nodes converge quickly and need not be re-computed over and over. Note that the latter advantage was also a key feature of some of the most important work of Kamvar and Haveliwala, as seen in [30] and [31].

2.2.6 Distributed randomized methods

In 2010, Ishii and Tempo introduced a distributed randomized approach for computing PageRank [24]. As the name suggests, this method involves the distributed (across all pages in the web) updating of the PageRank vector, where the individual page initiating the

update procedure at each time step is selected by a random process that follows a discrete uniform probability distribution. The goal is to efficiently determine the PageRank value of each page in the web without a decision maker or any particular ordering of the pages. They observe that their approach may be compared to the multi-agent consensus problem, in which multiple agents compare values with their neighbors in order to obtain a consensus value. There are three key features of the distributed randomized method: each page computes its own PageRank value by communicating with neighbors connected by direct links (incoming or outgoing), pages make the decision to initiate communication at random times that are independent from page to page, and minimal computation is required for each page.

The basic protocol of the distributed update scheme is as follows: at time k, a page $i \in \mathcal{V} = \{1, \ldots, n\}$ initiates the PageRank update by sending its own current PageRank value to each of the pages that it links to and requesting current values from each of the pages that link to it. At any given time, the page that initiates the update action is determined in a random manner, specified by a random process $\theta_k \in \mathcal{V}$. If, at time k, θ_k takes value i, then page i initiates the update action by communicating with connected pages. The random variables $\{\theta_k, k = 1, 2, \ldots\}$ are assumed to be independent and identically distributed (i.i.d.), with probability distribution given, for each $i \in \mathcal{V}$ and $k = 1, 2, \ldots$, by $\mathbb{P}(\theta_k = i) = 1/n$.

In this distributed approach, convergence to the PageRank vector is achieved in the mean-square sense through the time-average of the estimates. The generalized case where multiple pages are allowed to update simultaneously (at random) is also considered and is shown to exhibit the same convergence properties.

Zhao, Chen, and Fang [51] use stochastic approximation techniques to prove the almost sure convergence of the distributed randomized PageRank estimates to the true PageRank value, for both the single-page and multi-page update schemes. By the boundedness of the time average of the estimates, this strong consistency implies the mean-square convergence that was established directly by Ishii and Tempo in [24].

In 2012, Ishii, Tempo, and Bai [26] combined the distributed randomized update scheme

with a web aggregation approach in order to boost computational efficiency. As in the related approaches presented in Section 2.2.5, the web is aggregated by assigning each page to one of a number of groups and then each group computes one value that is the sum of the values of the group members. The aggregation itself is performed by identifying pages that share the following properties: the pages are placed under the same host/server so that their values can be computed together, each group has a sufficiently large number of internal links (i.e. pages have more links within their own group than pointing at pages in other groups consisting of multiple pages), and group members are expected to have similar PageRank values.

2.2.7 Monte Carlo methods

In 2007, Avrachenkov, Litvak, Nemirovsky, and Osipova used a probabilistic Monte Carlo approach to develop several algorithms for computing PageRank [6]. The main idea behind these methods is to approximate the PageRank vector based on the behavior of a large number of simulated random walks on the web graph whose termination depends on the teleportation parameter, t. In particular, at each step, a given random walk will terminate at its current location with probability t; otherwise, with probability $1 - t$, it will continue randomly to a new page by following the hyperlink structure of the web. Once every random walk has terminated, the PageRank is estimated according to the frequency with which each page in the web has been visited.

One option is to simply set the PageRank of a page $i \in \mathcal{V}$ equal to the fraction of walks that end at i. To reduce variance, an alternative method is proposed that takes into account not only information about the last-visited page, but rather about all pages visited by each simulated random walk. In both cases, the algorithms are considered with random-start (i.e. all simulations are initialized randomly across all pages in the web) as well as cyclic-start (i.e. a fixed, equal number of simulations is initialized at each of the n pages in the web).

The Monte Carlo algorithms of Avrachenkov et al. are able to determine the PageRank

of relatively important pages after just one iteration. Thus, whereas the original power iteration for computing PageRank could still be carried out occasionally, in order to obtain accurate PageRank values for all pages in the web, a Monte Carlo method could be used continuously at all other times in order to keep the PageRank values of relatively important pages up-to-date.

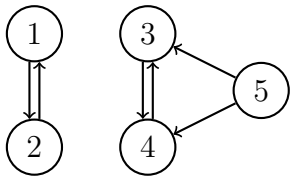
In 2013, Das Sarma, Molla, Pandurangan, and Upfal also applied the Monte Carlo approach to PageRank computation [49]. Based closely on the cyclic-start algorithms of Avrachenkov et al. in [6], they focus on distributed methods that execute rapidly via parallel implementation (i.e. letting multiple independent simulations be carried out simultaneously across multiple processors). Their most successful approach proceeds by simulating many short walks and later stitching them together to form longer walks, leading to an algorithm that yields a close approximation of the true PageRank while exhibiting a convergence rate that is sub-logarithmic in n.

2.2.8 Surveys

Over the years, a number of surveys have summarized the work done on the theory of PageRank. Examples include a 2004 paper by Langville and Meyer [35], which focused on the contributions of Kamvar and Haveliwala. Langville and Meyer also wrote a book [38] that provides a user-friendly introduction to the general theory and practice of search engine rankings (with an emphasis on the PageRank problem and its mathematical formulation) and includes MATLAB codes for many simulations relating to the computation of PageRank.

In 2006, Bryan and Leise [12] presented an overview of the original PageRank algorithm together with a discussion of its historical significance. In addition, this paper provides the following explicit example of the nonuniqueness of PageRank in a disconnected web where the teleportation model is not considered.

Example 2.2 (Nonunique PageRank)**.** Consider a small web of $n = 5$ pages described by the following web graph and hyperlink matrix:



$$H = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1/2 \\ 0 & 0 & 1 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The eigenspace of H is two-dimensional, with $(1/2, 1/2, 0, 0, 0)$ and $(0, 0, 1/2, 1/2, 0)$ forming one possible basis. Therefore, it is unclear which, if any, of these vectors or their linear combinations should be used for ranking the importance of pages in the web.

In 2011, Franceschet [16] described some major earlier works of the 20th century that influenced the inception of the PageRank concept, while also discussing relationships between PageRank and the fields of bibliometrics, sociometry, and econometrics. Most recently, Ishii and Tempo [25] provided an in-depth introduction to the PageRank problem and a summary of the work on it, with emphasis on the distributed randomized approach and web aggregation method presented in their earlier papers [23, 24, 26, 27]. They also address the application of PageRank theory to bibliometrics, specifically in the context of ranking control journals, and discuss relationships between PageRank and modern consensus problems.

2.3 Simulations

In this section, we present numerical simulations of the computation of PageRank via the power iteration. Our method for producing web graphs to use in the examples will be to employ a Pareto (power law) distribution to determine the number of incoming links of each webpage. In particular, all of the simulation examples in this section, as well as in Sections 3.4 and 4.5, will involve hyperlink matrices constructed via the following generative procedure.

For a web of size n, we begin with the vector

$$(1^p, 2^p, \ldots, (n-1)^p),$$

which is inverted component-wise and normalized so that the sum of its components is equal to one. The result,

$$\left(\frac{1^{-p}}{\sum_{i=1}^{n-1} i^{-p}}, \frac{2^{-p}}{\sum_{i=1}^{n-1} i^{-p}}, \ldots, \frac{(n-1)^{-p}}{\sum_{i=1}^{n-1} i^{-p}} \right),$$

is used to partition the interval $[0,1]$. In particular, we may write

$$\begin{aligned}
[0,1] &= \left[0, \frac{1^{-p}}{\sum_{i=1}^{n-1} i^{-p}}\right] \bigcup \left[\frac{1^{-p}}{\sum_{i=1}^{n-1} i^{-p}}, \frac{1^{-p}+2^{-p}}{\sum_{i=1}^{n-1} i^{-p}}\right] \bigcup \cdots \bigcup \left[\frac{\sum_{j=0}^{n-2} j^{-p}}{\sum_{i=1}^{n-1} i^{-p}}, 1\right] \\
&= \bigcup_{k=1}^{n-1} \left[\frac{\sum_{j=0}^{k-1} j^{-p}}{\sum_{i=1}^{n-1} i^{-p}}, \frac{\sum_{j=0}^{k} j^{-p}}{\sum_{i=1}^{n-1} i^{-p}}\right].
\end{aligned}$$

Next, for each each of the n pages in the simulated web graph, we generate a random number $r \in [0,1]$ and set the number of unique incoming links of that page equal to the integer $k \in \{1, \ldots, n-1\}$ such that

$$\frac{\sum_{j=0}^{k-1} j^{-p}}{\sum_{i=1}^{n-1} i^{-p}} < r \leq \frac{\sum_{j=0}^{k} j^{-p}}{\sum_{i=1}^{n-1} i^{-p}}.$$

Consequently, each page is assigned a number of in-links that ranges between 1 and $n-1$, with higher probability of a lower number due to the power law.

Taking care to avoid self-links and repeats, we randomly generate where each page's in-links come from. Finally, we back-link from any dangling nodes, as described in Section 2.1.1. The resulting hyperlink matrix $H \in \mathbb{R}^{n \times n}$ is sparse, nonnegative, and column-stochastic, as desired.

The choice of p directly impacts the link sparsity of the simulated web, with a higher power corresponding to greater sparsity. Various empirical studies, as summarized by Pandurangan, Raghavan, and Upfal in [46], have demonstrated that the value of $p = 2.1$ yields

graphs that accurately model the true structure of the World Wide Web. Therefore, we shall use this particular power in the following examples.

Example 2.3 (200-page Web)**.** We begin with an example of PageRank computation on a web of size $n = 200$. Using the power law distribution technique described above, we generate a web graph consisting of 200 vertices and 672 directed edges. There are two simple ways to visualize the structure of the this simulated web. The first involves a circular web graph diagram, while the second involves a sparsity plot of the hyperlink matrix, as illustrated in Figures 2.4 and 2.5, respectively.

In the case of Figure 2.4, the pages are ordered from 1 through 200 around the circle counterclockwise, with the lines between nodes representing links between pages. This diagram does not illustrate the directedness of the web graph, but it does allow us to easily identify areas of high link concentration, such as near Page 140.

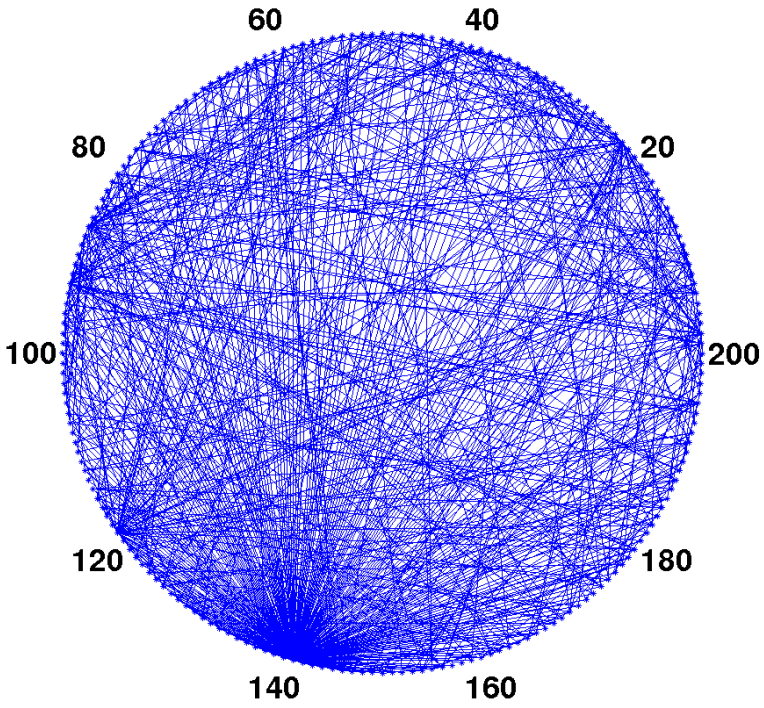


Figure 2.4: Circular web graph diagram for Ex. 2.3

Figure 2.5 consists of a 200-by-200 grid on which there is a dot at the point (i, j) if and only if there is a link from page j to page i. Thus, the striations in the plot allow us to easily identify pages with many incoming hyperlinks, i.e. pages that are likely to have high PageRank.

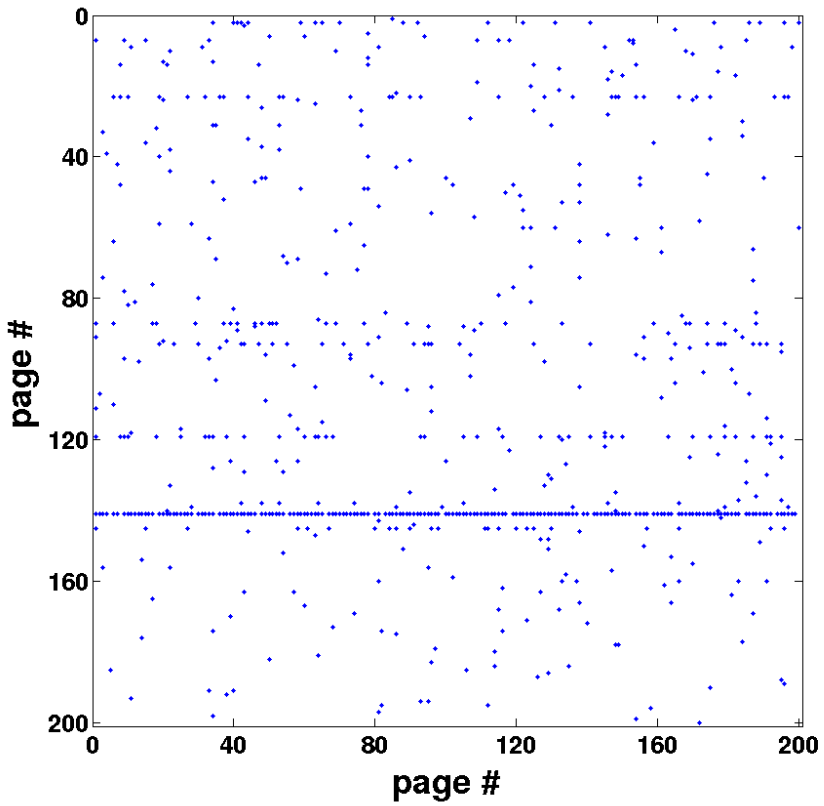


Figure 2.5: Hyperlink matrix sparsity plot for Ex. 2.3

Based on an examination of both Figure 2.4 and 2.5, we might suppose that Page 140 has the largest PageRank, followed by Page 119, and after that it becomes more difficult to determine.

Figure 2.6 illustrates the convergence of the PageRank iterates over the course of the first 20 iterations, while Figure 2.7 shows the converged PageRank values plotted against their corresponding page indices. Page 140 is indeed seen to be the one with largest PageRank, followed by Page 119, just as we conjectured from the link structure plot.

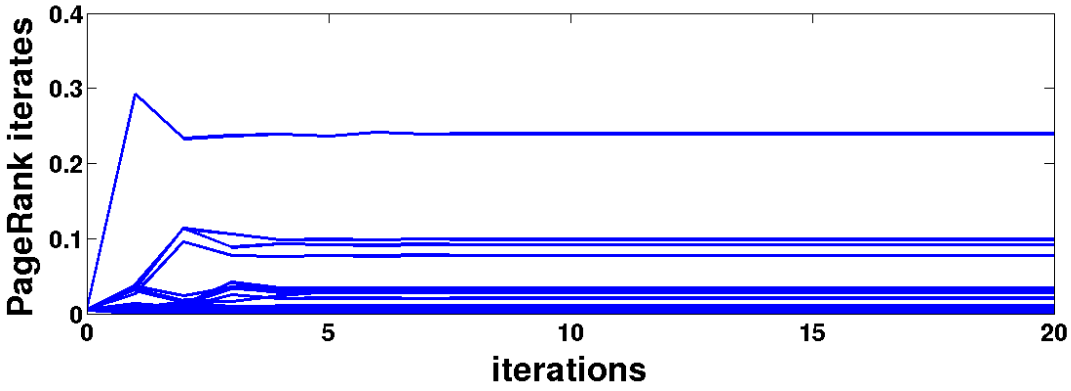


Figure 2.6: Convergence of PageRank values for Ex. 2.3

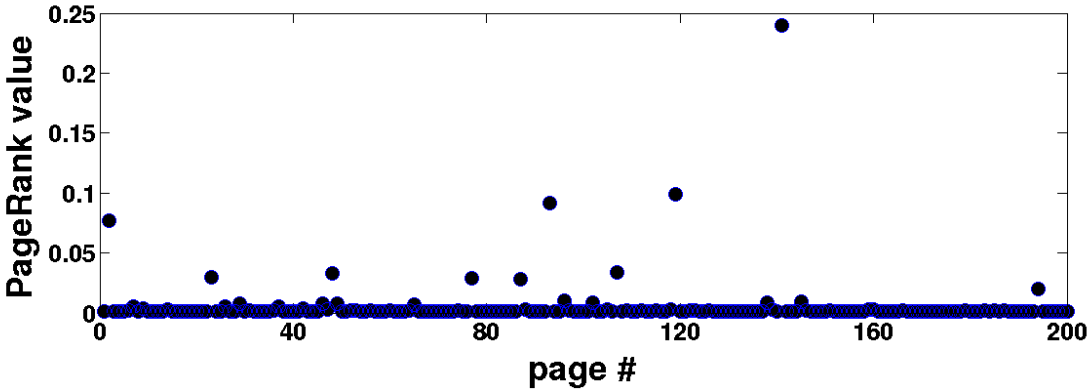


Figure 2.7: PageRank values plotted against page index for Ex. 2.3

Finally, in Figure 2.8, we plot the PageRank values against the number of incoming links of each page in the web. As expected, a greater number of in-links roughly corresponds to a higher PageRank value.

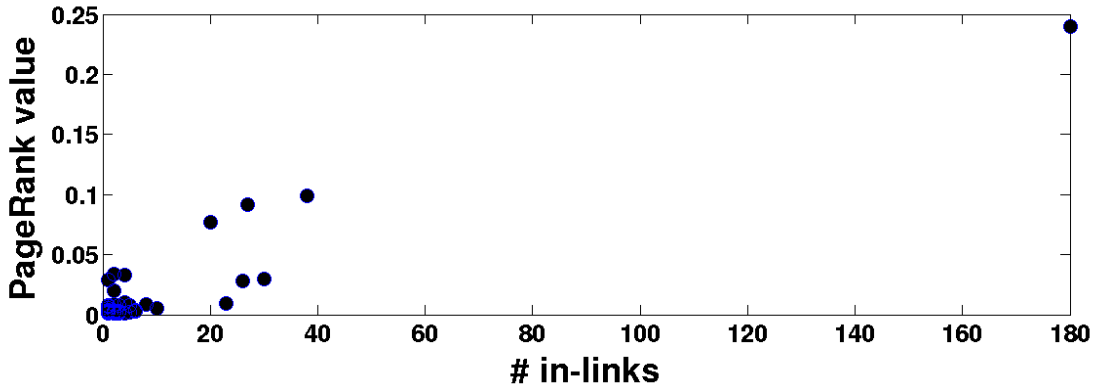


Figure 2.8: PageRank values plotted against number of in-links for Ex. 2.3

Example 2.4 (10000-page Web)**.** We next simulate PageRank computation on a web of size $n = 10000$. In this case, our generated graph consists of 706750 directed edges connecting the 10000 vertices. At this scale, a circular web graph diagram such as Figure 2.4 is useless (it appears simply as a completely filled-in circle), and even a hyperlink matrix sparsity plot (see Figure 2.11 below) is difficult to interpret. Instead, we proceed directly to applying power iteration, yielding Figures 2.9 and 2.10.

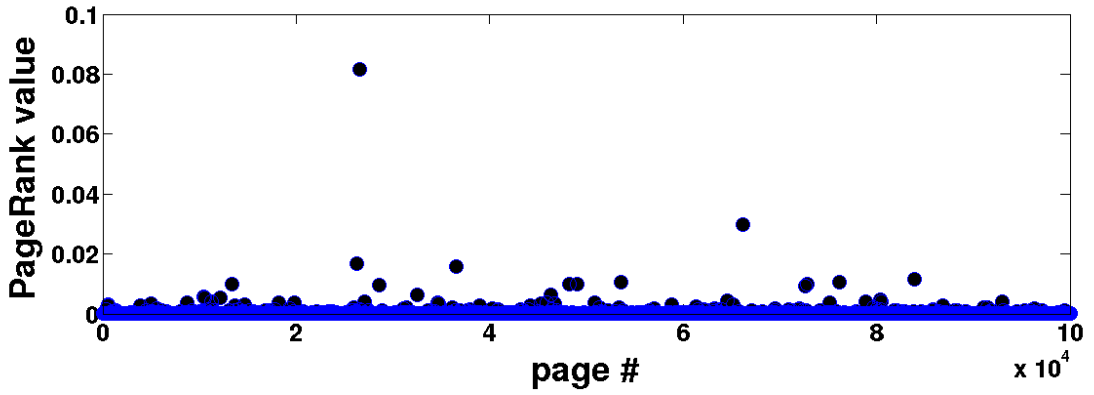


Figure 2.9: PageRank values plotted against page index for Ex. 2.4

In Figure 2.9, the converged PageRank values are plotted against page index. The page with highest PageRank value in this example turns out to be page 2712, followed by page 6690. In Figure 2.10, the PageRank values are plotted against the number of incoming links of each page in the web.

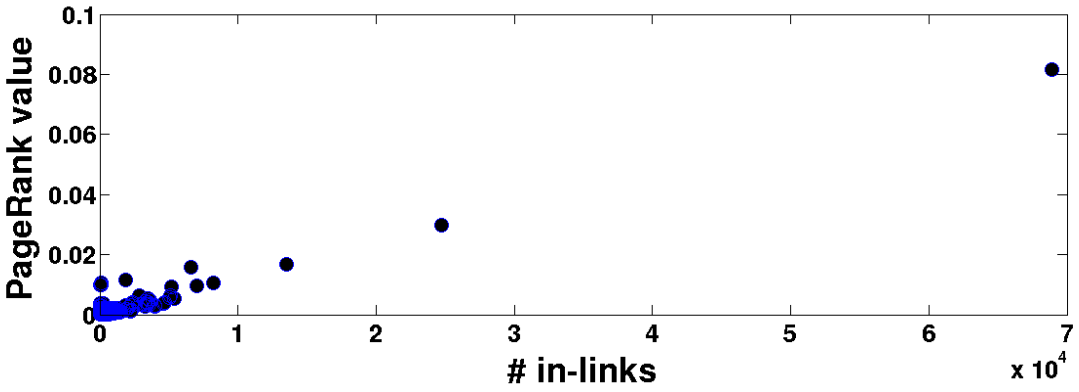


Figure 2.10: PageRank values plotted against number of in-links for Ex. 2.4

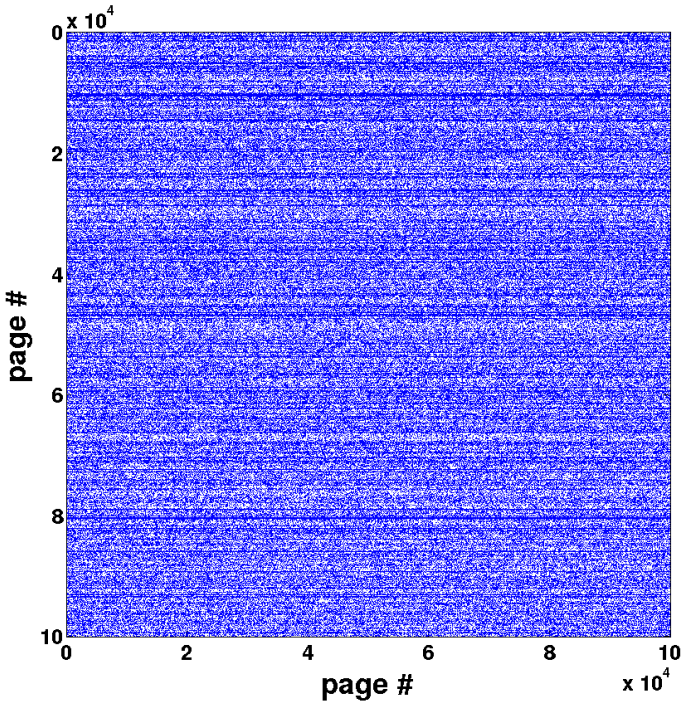


Figure 2.11: Hyperlink matrix sparsity plot for Ex. 2.4

Chapter 3

Parameter Estimation in a Stochastic System Model for PageRank

In this chapter, we present a model for PageRank whose dynamics are described by a controlled stochastic system that depends on an unknown parameter. The fact that the value of the parameter is unknown implies that the system is unknown. We establish strong consistency of a least squares estimator for the parameter. Furthermore, motivated by the recent work of Ishii and Tempo on distributed randomized methods for PageRank computation [24], we show that the least squares estimator remains strongly consistent within a distributed framework. For more on parameter estimation in stochastic systems, we refer to the texts of Chen and Guo [13], Kumar and Varaiya [33], as well as the monograph of Pasik-Duncan [47] and the references listed therein.

The remainder of this chapter is organized as follows. In Section 3.1, we introduce a stochastic system formulation of PageRank. In Sections 3.2 and 3.3, we prove the strong consistency of the least squares estimator of an unknown parameter in the system, in the centralized and distributed cases, respectively. Finally, in Section 3.4, we present numerical results that support the theory regarding the strong consistency of the least squares estimator.

3.1 A Stochastic System Model for PageRank

Let us consider a model for PageRank whose dynamics are described by the discrete-time, controlled, stochastic, linear system

$$X_{j+1} = TX_j + SU_j + \xi_{j+1},\ j = 0, 1, \ldots, \tag{3.1}$$

where X_j and U_j are, respectively, an n-vector denoting the state of the system and an m-vector denoting the control signal at time j. The matrices T and S are constant, of size $n \times n$ and $n \times m$, respectively, and U_j has the form

$$U_j = K_j X_j,\ j = 0, 1, \ldots,$$

where $K_j = K_j(X_0, \ldots, X_j, U_0, \ldots, U_{j-1})$, $j = 0, 1, \ldots$, are $m \times n$ matrices. The sequence $\{K_j, j = 0, 1, \ldots\}$ is the *control policy* for the system and if $K_j = K$ for all $j = 0, 1, \ldots$, the control policy is said to be *stationary*. The perturbations $\{\xi_j, j = 0, 1, \ldots\}$ are independent and identically distributed (i.i.d.) random n-vectors with zero mean, covariance matrix Q, and finite fourth moment. The initial state X_0 is assumed to be non-random.

We shall say that the process $\{X_j, j = 0, 1, \ldots\}$ is *stable* under the control policy $\{K_j, j = 0, 1, \ldots\}$, and that the control is also stable, if

$$\limsup_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} \mathbb{E}_x \|X_j\|^2 \le C, \tag{3.2}$$

where $\mathbb{E}_x$ denotes expectation conditional on the event $X_0 = x$ and C is a finite constant independent of x.

Henceforth, we let $\{\mathcal{F}_j, j = 0, 1, \ldots\}$ denote the natural filtration for $\{X_j, j = 0, 1, \ldots\}$; that is, we set $\mathcal{F}_n$ as the σ-algebra generated by $\{X_j, j = 0, 1, \ldots, n\}$ for $n = 0, 1, \ldots$.

3.1.1 Quadratic forms and their expectations.

If v is a random n-vector and Λ is an $n \times n$ symmetric matrix, then $v'\Lambda v$ is a scalar quantity called a *quadratic form* in v. The following lemma establishes a formula for the expectation of a quadratic form.

Lemma 3.1.1. *For v and Λ defined as above, we have $\mathbb{E}[v'\Lambda v] = \mathrm{tr}(\Lambda\Sigma) + \mu'\Lambda\mu$, where μ and Σ are the mean and covariance matrix, respectively, of v.*

Proof. By the linearity of expectation and trace, and the invariance of trace under cyclic permutations, we have

$$\mathbb{E}\left[\mathrm{tr}(v'\Lambda v)\right] = \mathbb{E}\left[\mathrm{tr}(\Lambda vv')\right] = \mathrm{tr}\left(\Lambda\,\mathbb{E}[vv']\right) = \mathrm{tr}\left(\Lambda\Sigma + \Lambda\mu\mu'\right) = \mathrm{tr}(\Lambda\Sigma) + \mathrm{tr}(\mu'\Lambda\mu).$$

This completes the proof, since $v'\Lambda v$ is a scalar quantity that is equal to its own trace. □

As a special case of this result, we observe that

$$\mathbb{E}\left[\xi_j' W \xi_j\right] = \mathrm{tr}(WQ),\ \ j = 0, 1, \ldots,$$

where W is any $n \times n$ symmetric matrix and $\{\xi_j, j = 0, 1, \ldots\}$ are the perturbations in the system (3.1). This fact will be used in the proofs of Theorems 3.2.2 and 3.3.1, as well as in Chapter 4.

3.2 Estimation of an Unknown Parameter

Motivated by the early work of Mandl [43] and Pasik-Duncan [47], let us now proceed to consider the case in which the dynamics of our model are described by a system that is unknown. We suppose that system's behavior depends on an unknown parameter, and the fact that the value of the parameter is unknown implies that the system is unknown. In

particular, whereas the matrix S in (3.1) remains known, the matrix T is replaced by

$$T(\alpha) = (1-t)H(\alpha) + \frac{t}{n}\mathbf{1}\mathbf{1}',$$

where $H(\alpha)$ has the affine form

$$H(\alpha) = F^{(0)} + \alpha^{(1)}F^{(1)} + \cdots + \alpha^{(q)}F^{(q)},$$

with $F^{(0)}, F^{(1)}, \ldots, F^{(q)}$ being known matrices and $\alpha = (\alpha^{(1)}, \ldots, \alpha^{(q)})$ a real-valued unknown parameter with true value $\alpha_0 = \left(\alpha_0^{(1)}, \ldots, \alpha_0^{(q)}\right)$. The set $\mathcal{A}_\alpha \subset \mathbb{R}^q$ of possible values of α is assumed to be open and bounded. For simplicity of notation, we let $H = H(\alpha)$, $H^0 = H(\alpha_0)$, $T = T(\alpha)$, and $T^0 = T(\alpha_0)$.

We shall consider online least squares identification of the unknown parameter α; that is, the least squares estimation of α_0 that proceeds simultaneously with the evolution of the system. For $k = 1, 2, \ldots$, the estimate $\hat{\alpha}_k$ of α_0 is obtained by minimizing the sum of squares

$$\mathcal{L}_k(\alpha) = \sum_{j=0}^{k-1} (X_{j+1} - TX_j - SU_j)' R (X_{j+1} - TX_j - SU_j) \tag{3.3}$$

with respect to α over the compact set $\overline{\mathcal{A}}_\alpha$, where $R \geq 0$ is an $n \times n$ matrix chosen by the controller.

First, we will establish the strong consistency of this parameter estimation method. Then, in Chapter 4, we will introduce adaptive control policies of the form

$$K_j = \left(T(\alpha_j^*)\right), \ j = 0, 1, \ldots,$$

where $\{\alpha_j^*, j = 0, 1, \ldots\}$ is closely related to the sequence of least squares estimates.

We introduce the set $\mathcal{K}_\alpha = \{K(\alpha) : \alpha \in \mathcal{A}_\alpha\}$. Only matrices from $\mathcal{K}_\alpha$ will be employed for the control. The following hypothesis guarantees the stability of the system, regardless of

the rule used to interchange the controls from $\mathcal{K}_\alpha$.

Assumption 3.1. *For each $\alpha \in \mathcal{A}_\alpha$, an $n \times n$ matrix $V > 0$ can be found such that*

$$V \geq I + (T(\alpha) + SK)' V (T(\alpha) + SK) \tag{3.4}$$

for all $K \in \mathcal{K}_\alpha$.

The following stability lemma, due to Mandl [43], will be used in the proof of Theorem 3.2.2, as well as Theorem 3.3.1.

Lemma 3.2.1. *The process $\{X_j, j = 0, 1, \dots\}$ is stable under a control policy $\{K_j, j = 0, 1, \dots\}$ taking values in $\mathcal{K}_\alpha$. Moreover, we have*

$$\sum_{k=1}^{\infty} k^{-2} \mathbb{E}\|X_k\|^4 < \infty. \tag{3.5}$$

Proof. Suppose that V satisfies Assumption 3.1 with $\alpha = \alpha_0$. Then we have

$$\begin{aligned}
\|X_j\|^2 + \mathbb{E}_x \left[X_{j+1}' V X_{j+1} \mid \mathcal{F}_j \right] &= \|X_j\|^2 + \mathbb{E}_x \left[\left(T^0 X_j + S U_j + \xi_{j+1} \right)' V \left(T^0 X_j + S U_j + \xi_{j+1} \right) \mid \mathcal{F}_j \right] \\
&= \|X_j\|^2 + X_j' \left(T^0 + S K_j \right)' V \left(T^0 + S K_j \right) X_j + \mathbb{E}_x \left[\xi_{j+1}' V \xi_{j+1} \mid \mathcal{F}_j \right] \\
&= \|X_j\|^2 + X_j' \left(T^0 + S K_j \right)' V \left(T^0 + S K_j \right) X_j + \operatorname{tr}(VQ) \\
&\leq X_j' V X_j + \operatorname{tr}(VQ),
\end{aligned}$$

so that

$$\mathbb{E}_x \|X_j\|^2 + \mathbb{E}_x \left[X_{j+1}' V X_{j+1} \right] \leq \mathbb{E}_x \left[X_j' V X_j \right] + \operatorname{tr}(VQ), \ j = 0, 1, \dots.$$

Hence, stability of the process follows from the fact that

$$\sum_{j=0}^{k-1} \mathbb{E}_x \|X_j\|^2 + \mathbb{E}_x \left[X_k' V X_k \right] \leq \mathbb{E}_x \left[X_0' V X_0 \right] + k \operatorname{tr}(VQ), \ k = 1, 2, \dots.$$

To establish (3.5), we proceed in a similar manner. Assumption 3.1 implies that

$$(X_0' V X_0)^2 \geq \|X_0\|^4 + \left(X_0' \left(T^0 + SK\right)' V \left(T^0 + SK\right) X_0\right)^2$$

for all $K \in \mathcal{K}_\alpha$. Hence,

$$\mathbb{E}\|X_j\|^4 + \mathbb{E}\left[\left(X_{j+1}' V X_{j+1}\right)^2\right] \leq \mathbb{E}\left[\left(X_j' V X_j\right)^2\right] + C\left(1 + \mathbb{E}\left[\|X_{j+1}\|^2\right]\right), \ j = 0, 1, \ldots.$$

From these inequalities, it follows that

$$\limsup_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} \mathbb{E}\|X_j\|^4 \leq C\left(1 + \limsup_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} \mathbb{E}\|X_j\|^2\right),$$

which implies (3.5). □

Before establishing the strong consistency of the least squares estimates for the true unknown parameter value α_0, we impose an additional condition.

Assumption 3.2. *Let $\tilde{\alpha}, \alpha \in \overline{\mathcal{A}}_\alpha$. Then there exists an $n \times n$ matrix*

$$J(\tilde{\alpha}, \alpha) = \sum_{i,j} \left(\tilde{\alpha}^{(i)} - \alpha^{(i)}\right)\left(\tilde{\alpha}^{(j)} - \alpha^{(j)}\right) J_{ij}(\alpha) \geq 0$$

such that

$$J(\tilde{\alpha}, \alpha) \leq \left(T(\tilde{\alpha}) - T(\alpha)\right)' R \left(T(\tilde{\alpha}) - T(\alpha)\right) + \left(T(\alpha) + SK\right)' J(\tilde{\alpha}, \alpha) \left(T(\alpha) + SK\right) \tag{3.6}$$

for all $K \in \mathcal{K}_\alpha$, and

$$\operatorname{tr}\left(J(\tilde{\alpha}, \alpha) Q\right) \geq \lambda \|\tilde{\alpha} - \alpha\|^2, \tag{3.7}$$

where $\lambda > 0$.

This second assumption holds whenever

$$\operatorname{tr}\left[(T(\tilde{\alpha}) - T(\alpha))' R (T(\tilde{\alpha}) - T(\alpha)) Q\right] \geq \lambda \|\tilde{\alpha} - \alpha\|^2.$$

We are now prepared to state and prove our first result.

Theorem 3.2.2. *Under any control policy $\{K_j, j = 0, 1, \dots\}$ taking values in $\mathcal{K}_\alpha$, the sequence of least squares estimates $\{\hat{\alpha}_k, k = 1, 2, \dots\}$, obtained by minimizing the sum of squares* (3.3) *over $\overline{\mathcal{A}}_\alpha$, is strongly consistent for $\alpha_0 \in \mathcal{A}_\alpha$. That is, $\hat{\alpha}_k \xrightarrow{\text{a.s.}} \alpha_0$ as $k \to \infty$.*

Proof. First, observe that $\xi'_{j+1} R (H - H^0) X_j$ is a scalar, so it is equal to its transpose, $X'_j (H - H^0)' R \xi_{j+1}$. By the definition of the system (3.1), we have

$$\begin{aligned}
\mathcal{L}_k(\alpha) &= \sum_{j=0}^{k-1} \left(T^0 X_j + S U_j + \xi_{j+1} - [T X_j + S U_j]\right)' R \left(T^0 X_j + S U_j + \xi_{j+1} - [T X_j + S U_j]\right) \\
&= \sum_{j=0}^{k-1} \left((1-t) H^0 X_j + \frac{t}{n}\mathbf{1} + S U_j + \xi_{j+1} - \left[(1-t) H X_j + \frac{t}{n}\mathbf{1} + S U_j\right]\right)' \\
&\qquad \cdot R \left((1-t) H^0 X_j + \frac{t}{n}\mathbf{1} + S U_j + \xi_{j+1} - \left[(1-t) H X_j + \frac{t}{n}\mathbf{1} + S U_j\right]\right) \\
&= \sum_{j=0}^{k-1} \left(\xi_{j+1} - (1-t)\left(H - H^0\right) X_j\right)' R \left(\xi_{j+1} - (1-t)\left(H - H^0\right) X_j\right) \\
&= \sum_{j=0}^{k-1} \left(\xi'_{j+1} - (1-t) X'_j \left(H - H^0\right)'\right) R \left(\xi_{j+1} - (1-t)\left(H - H^0\right) X_j\right) \\
&= \sum_{j=0}^{k-1} \xi'_{j+1} R \xi_{j+1} + (1-t)^2 \sum_{j=0}^{k-1} X'_j \left(H - H^0\right)' R \left(H - H^0\right) X_j \\
&\qquad - 2(1-t) \sum_{j=0}^{k-1} \xi'_{j+1} R \left(H - H^0\right) X_j.
\end{aligned}$$

Since

$$\sum_{j=0}^{k-1} \xi'_{j+1} R \xi_{j+1} = \mathcal{L}_k(\alpha_0),$$

we may write

$$\mathcal{L}_k(\alpha) - \mathcal{L}_k(\alpha_0) = (1-t)^2 \sum_{j=0}^{k-1} X_j' \left(H - H^0\right)' R \left(H - H^0\right) X_j$$
$$- 2(1-t) \sum_{j=0}^{k-1} \xi_{j+1}' R \left(H - H^0\right) X_j. \tag{3.8}$$

Let the sequence $\{Y_j, j = 0, 1, \dots\}$ be defined by

$$Y_j = 2(1-t)\, \xi_{j+1}' R \left(H - H^0\right) X_j, \quad j = 0, 1, \dots.$$

Then, since

$$\mathbb{E}\left[Y_j \mid \mathcal{F}_j\right] = 2(1-t)\, \mathbb{E}\left[\xi_{j+1}' \mid \mathcal{F}_j\right] R \left(H - H^0\right) X_j$$
$$= 0, \quad \text{a.s.}, \ j = 0, 1, \dots,$$

we see that the Y_j, $j = 0, 1, \dots$, are martingale differences. Furthermore,

$$\mathbb{E}\left[Y_j^2 \mid \mathcal{F}_j\right] = 4(1-t)^2 X_j' \left(H - H^0\right)' R\, \mathbb{E}\left[\xi_{j+1}\xi_{j+1}' \mid \mathcal{F}_j\right] R \left(H - H^0\right) X_j$$
$$= 4(1-t)^2 X_j' \left(H - H^0\right)' RQR \left(H - H^0\right) X_j, \quad \text{a.s.}, \ j = 0, 1, \dots.$$

From Lemma 3.2.1, it follows that

$$\sum_{k=0}^{\infty} (k+1)^{-2}\, \mathbb{E} Y_k^2 < \infty.$$

Hence, by Lemma B.0.1, we have

$$\lim_{k \to \infty} k^{-1} \sum_{j=0}^{k-1} Y_j = 0, \quad \text{a.s.}$$

By definition of Y_j, the expression on the left-hand side above is a linear form in $\alpha - \alpha_0$ that

converges to 0 almost surely, for arbitrary α, as $k \to \infty$. Indeed, since

$$\begin{aligned} Y_j &= 2(1-t)\,\xi'_{j+1} R\left(H - H^0\right) X_j \\ &= 2(1-t)\,\xi'_{j+1} R\left(\left[F^{(0)} + \alpha^{(1)}F^{(1)} + \cdots + \alpha^{(q)}F^{(q)}\right] - \left[F^{(0)} + \alpha_0^{(1)}F^{(1)} + \cdots + \alpha_0^{(q)}F^{(q)}\right]\right) X_j \\ &= \sum_{i=1}^{q} 2(1-t)\left(\alpha^{(i)} - \alpha_0^{(i)}\right) \xi'_{j+1} R F^{(i)} X_j \end{aligned}$$

we obtain

$$\begin{aligned} \sum_{j=0}^{k-1} Y_j &= \sum_{j=0}^{k-1}\sum_{i=1}^{q} 2(1-t)\left(\alpha^{(i)} - \alpha_0^{(i)}\right) \xi'_{j+1} R F^{(i)} X_j \\ &= \sum_{i=1}^{q}\left(\alpha^{(i)} - \alpha_0^{(i)}\right)\left[\sum_{j=0}^{k-1} 2(1-t)\,\xi'_{j+1} R F^{(i)} X_j\right] \\ &= \sum_{i=1}^{q}\left(\alpha^{(i)} - \alpha_0^{(i)}\right)\ell_i(k), \end{aligned}$$

where

$$\ell_i(k) = \sum_{j=0}^{k-1} 2(1-t)\,\xi'_{j+1} R F^{(i)} X_j.$$

Hence,

$$k^{-1}\sum_{j=0}^{k-1} Y_j = k^{-1}\sum_{i=1}^{q}\left(\alpha^{(i)} - \alpha_0^{(i)}\right)\ell_i(k) \tag{3.9}$$

and it follows that

$$\lim_{k\to\infty} k^{-1}\|\ell(k)\| = 0, \quad \text{a.s.} \tag{3.10}$$

We next consider the first sum on the right-hand side of (3.8). Set, for $J = J(\alpha, \alpha_0)$,

$$Z_j = X'_{j+1} J X_{j+1} - X'_j J X_j + X'_j\left(T - T^0\right)' R\left(T - T^0\right) X_j - \operatorname{tr}(JQ),\ j = 0, 1, \ldots.$$

From (3.6), it follows that

$$\begin{aligned}
\mathbb{E}\left[Z_j \mid \mathcal{F}_j\right] &= \mathbb{E}\left[X_{j+1}' J X_{j+1} \mid \mathcal{F}_j\right] - X_j' J X_j + X_j' \left(T - T^0\right)' R \left(T - T^0\right) X_j - \operatorname{tr}(JQ) \\
&= X_j'(T + SK)' J (T + SK) X_j + \mathbb{E}\left[\xi_{j+1}' J \xi_{j+1} \mid \mathcal{F}_j\right] \\
&\quad - X_j' J X_j + X_j' \left(T - T^0\right)' R \left(T - T^0\right) X_j - \operatorname{tr}(JQ) \\
&= X_j'(T + SK)' J (T + SK) X_j - X_j' J X_j + X_j' \left(T - T^0\right)' R \left(T - T^0\right) X_j \\
&\geq 0, \ j = 0, 1, \ldots . \qquad (3.11)
\end{aligned}$$

Next, we introduce the martingale

$$M_k = \sum_{j=0}^{k-1} \left(Z_j - \mathbb{E}\left[Z_j \mid \mathcal{F}_j\right]\right), \ k = 1, 2, \ldots .$$

From (3.11) and the fact that

$$\begin{aligned}
T - T^0 &= \left[(1-t)H + \frac{t}{n}\mathbf{1}\mathbf{1}'\right] - \left[(1-t)H^0 + \frac{t}{n}\mathbf{1}\mathbf{1}'\right] \\
&= (1-t)\left(H - H^0\right)
\end{aligned}$$

it follows that

$$\begin{aligned}
M_k &\leq \sum_{j=0}^{k-1} Z_j \\
&= \sum_{j=0}^{k-1} \left[X_{j+1}' J X_{j+1} - X_j' J X_j + (1-t)^2 X_j' \left(H - H^0\right)' R \left(H - H^0\right) X_j - \operatorname{tr}(JQ)\right] \\
&= X_k' J X_k - k \operatorname{tr}(JQ) + (1-t)^2 \sum_{j=0}^{k-1} X_j' \left(H - H^0\right)' R \left(H - H^0\right) X_j, \ k = 1, 2, \ldots .
\end{aligned}$$

Hence, by (3.7), we have

$$k^{-1}M_k - k^{-1}X_k'JX_k + \lambda\|\alpha - \alpha_0\|^2$$
$$\leq k^{-1}(1-t)^2 \sum_{j=0}^{k-1} X_j' \left(H - H^0\right)' R \left(H - H^0\right) X_j, \ k = 1, 2, \ldots. \tag{3.12}$$

Lemma 3.2.1 implies that Lemma B.0.1 holds for $\{M_k, k = 1, 2, \ldots\}$ and that

$$\lim_{k\to\infty} k^{-1}X_k'JX_k = 0, \quad \text{a.s.}$$

The sum of the first two terms on the left-hand side of (3.12) is a quadratic form in $\alpha - \alpha_0$ that converges to 0 almost surely, for arbitrary α, as $k \to \infty$. Consequently, there exists an almost surely finite random variable N with the property that

$$k^{-1}M_k - k^{-1}X_k'JX_k \geq -\frac{\lambda}{2}\|\alpha - \alpha_0\|^2 \tag{3.13}$$

for all α and for $k \geq N$. By (3.8) and (3.9), we have

$$k^{-1}\left(\mathcal{L}_k(\alpha) - \mathcal{L}_k\left(\alpha_0\right)\right) = k^{-1}(1-t)^2 \sum_{j=0}^{k-1} X_j' \left(H - H^0\right)' R \left(H - H^0\right) X_j - k^{-1} \sum_{i=1}^{q} \left(\alpha^{(i)} - \alpha_0^{(i)}\right) \ell_i(k).$$

Hence, it follows from (3.12) and (3.13) that

$$k^{-1}\left(\mathcal{L}_k(\alpha) - \mathcal{L}_k\left(\alpha_0\right)\right) \geq \frac{\lambda}{2}\|\alpha - \alpha_0\|^2 - k^{-1}\|\alpha - \alpha_0\| \, \|\ell(k)\|$$

for all α whenever $k \geq N$. Since $\mathcal{L}_k(\hat{\alpha}_k) = \min_{\alpha} \mathcal{L}_k(\alpha)$, we have

$$\mathcal{L}_k(\hat{\alpha}_k) - \mathcal{L}_k\left(\alpha_0\right) \leq 0,$$

and thus

$$\frac{\lambda}{2}\|\hat{\alpha}_k - \alpha_0\| \leq k^{-1}\|\ell(k)\| \tag{3.14}$$

for $k \geq N$. Hence, by (3.10), we conclude that

$$\lim_{k\to\infty} \|\hat{\alpha}_k - \alpha_0\| = 0, \quad \text{a.s.},$$

as desired. □

Fix $\alpha_0^* \in \mathcal{A}_\alpha$ and, for $j = 1, 2, \ldots$, define

$$\alpha_j^* = \begin{cases} \hat{\alpha}_j, & \text{if } \hat{\alpha}_j \in \mathcal{A}_\alpha; \\ \alpha_0^*, & \text{otherwise.} \end{cases} \tag{3.15}$$

Since α_0 is contained in the open set $\mathcal{A}_\alpha$, it follows from Theorem 3.2.2 that, with probability one, $\hat{\alpha}_j$ must always lie within $\mathcal{A}_\alpha$ for j large enough. In particular, we have the following corollary.

Corollary 3.2.3. *Under any control policy $\{K_j, j = 0, 1, \ldots\}$ taking values in $\mathcal{K}_\alpha$, we have* $\lim_{j\to\infty} \alpha_j^* = \alpha_0$ a.s.

3.3 Distributed Least Squares Estimation

In Section 2.2.6, we introduced the recent work of Ishii and Tempo on a distributed randomized approach to PageRank computation [24]. We now revisit this method in the context of the stochastic system model for PageRank. Recall that the basic protocol for the distributed randomized approach is as follows: at time k, a page $i \in \mathcal{V} = \{1, \ldots, n\}$ initiates the PageRank update by sending its own current PageRank value to each of the pages that it links to and requesting current values from each of the pages that link to it. At any given time, the page that initiates the update action is determined in a random manner, specified by a random process $\theta_k \in \mathcal{V}$. If, at time k, θ_k takes value i, then page i initiates the update action by communicating with connected pages. The random variables $\{\theta_k, k = 1, 2, \ldots\}$ are assumed to be i.i.d. with probability distribution given, for each $i \in \mathcal{V}$ and $k = 1, 2, \ldots$, by

$\mathbb{P}(\theta_k = i) = 1/n$.

The first step in the distributed randomized approach is to form the distributed hyperlink matrices, H_i, $i = 1, \ldots, n$ (one for each page in the web), from the original hyperlink matrix, H. These matrices are constructed so that the ith row and column of H_i coincide with those of H, while the remaining diagonal elements are selected so that H_i is column stochastic and all other elements are zero.

Next, as in the case of the original PageRank algorithm, the distributed link matrices are modified in order to guarantee the uniqueness of the PageRank vector. The distributed teleportation matrices, T_i, are defined by

$$T_i = \left(1 - \tilde{t}\right) H_i + \frac{\tilde{t}}{n} \mathbf{1}\mathbf{1}',$$

where $\tilde{t} \in (0, 1)$ is a modified teleportation parameter that replaces t. We will assume that $\tilde{t} = 2t/(n - t(n - 2))$, as this value was shown by Ishii and Tempo in [24] to ensure the mean-square convergence of the distributed update scheme to the true PageRank vector.

We now proceed to consider the stochastic system formulation of the PageRank problem within the distributed framework. We will show that strong consistency of least squares estimator of an unknown parameter in the link matrix of the stochastic PageRank system is achieved even in the case of distributed randomized updating.

As in the Section 3.2, let α be a q-dimensional unknown parameter with true value α_0. With $H(\alpha)$ defined as before, let the α-dependent distributed link matrices be given by

$$H_{\theta_j}(\alpha) = F_{\theta_j}^{(0)} + \alpha^{(1)} F_{\theta_j}^{(1)} + \cdots + \alpha^{(q)} F_{\theta_j}^{(q)},$$

for $j = 0, 1, \ldots,$ where

$$\left(F_{\theta_j}^{(0)}\right)_{k\ell} = \begin{cases} \left(F^{(0)}\right)_{k\ell}, & \text{if } \theta_j = k \text{ or } \theta_j = \ell, \\ 1 - \left(F^{(0)}\right)_{\theta_j \ell}, & \text{if } k = \ell \neq \theta_j, \\ 0, & \text{otherwise,} \end{cases}$$

and

$$\left(F_{\theta_j}^{(i)}\right)_{k\ell} = \begin{cases} \left(F^{(i)}\right)_{k\ell}, & \text{if } \theta_j = k \text{ or } \theta_j = \ell, \\ -\left(F^{(i)}\right)_{\theta_j \ell}, & \text{if } k = \ell \neq \theta_j, \\ 0, & \text{otherwise,} \end{cases}$$

for $i = 1, \ldots, q$. By construction, the matrices $H_{\theta_j}(\alpha)$, $j = 0, 1, \ldots,$ retain a familiar structure: the θ_jth row and column coincide with those of $H(\alpha)$, the remaining diagonal elements are chosen such that $H_{\theta_j}(\alpha)$ is a column stochastic matrix, and all other elements are zero. Let the α-dependent teleportation matrices be given by

$$T_{\theta_j}(\alpha) = \left(1 - \tilde{t}\right) H_{\theta_j}(\alpha) + \frac{\tilde{t}}{n} \mathbf{1}\mathbf{1}'.$$

For simplicity of notation, we let $H_{\theta_j} = H_{\theta_j}(\alpha)$, $H_{\theta_j}^0 = H_{\theta_j}(\alpha_0)$, $T_{\theta_j} = T_{\theta_j}(\alpha)$, and $T_{\theta_j}^0 = T_{\theta_j}(\alpha_0)$.

Example 3.1 (Distributed Hyperlink Matrices)**.** Consider a simple web of $n = 3$ pages whose original hyperlink matrix is

$$H = \begin{bmatrix} 0 & 0 & 1 \\ 1/2 & 0 & 0 \\ 1/2 & 1 & 0 \end{bmatrix}.$$

Suppose that

$$F^{(0)} = \begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} \qquad \text{and} \qquad F^{(1)} = \begin{bmatrix} 2 & 0 & 0 \\ 1 & 0 & 2 \\ 1 & 2 & 0 \end{bmatrix}$$

are known matrices, and α is a one-dimensional parameter with true value $\alpha_0 = 1/2$. Then

$$H = F^{(0)} + \alpha_0 F^{(1)}$$

and the distributed link matrices are given by

$$H_1 := \begin{bmatrix} 0 & 0 & 1 \\ 1/2 & 1 & 0 \\ 1/2 & 0 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} 2 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} =: F_1^{(0)} + \alpha_0 F_1^{(1)},$$

$$H_2 := \begin{bmatrix} 1/2 & 0 & 0 \\ 1/2 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 2 \\ 0 & 2 & -2 \end{bmatrix} =: F_2^{(0)} + \alpha_0 F_2^{(1)},$$

$$H_3 := \begin{bmatrix} 1/2 & 0 & 1 \\ 0 & 0 & 0 \\ 1/2 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} -1 & 0 & 0 \\ 0 & -2 & 2 \\ 1 & 2 & 0 \end{bmatrix} =: F_3^{(0)} + \alpha_0 F_3^{(1)}.$$

We now consider the least squares estimation of α_0 within the distributed framework. For $k = 1, 2, \ldots,$ the estimate $\hat{\alpha}_k$ of α_0 is obtained by minimizing the sum of squares

$$\mathcal{L}_k(\alpha) = \sum_{j=0}^{k-1} \left(X_{j+1} - T_{\theta_j} X_j - S U_j \right)' R \left(X_{j+1} - T_{\theta_j} X_j - S U_j \right) \tag{3.16}$$

with respect to α over the compact set $\overline{\mathcal{A}}_\alpha$, where again $R \geq 0$ is an $n \times n$ matrix chosen by the controller.

In this distributed setting, our Assumptions 3.1 and 3.2 must be appropriately modified;

in particular, we now require that they hold when the original teleportation matrix $T(\alpha)$ is replaced by the distributed teleportation matrices $T_{\theta_j}(\alpha)$ for all $\theta_j \in \mathcal{V}$.

Indeed, our first hypothesis replaces Assumption 3.1 and ensures that Lemma 3.2.1 can be applied as before.

Assumption 3.3. *For each $\alpha \in \mathcal{A}_\alpha$ and $\theta_j \in \mathcal{V}$, an $n \times n$ matrix $V > 0$ can be found such that*

$$V \geq I + \left(T_{\theta_j}(\alpha) + SK\right)' V \left(T_{\theta_j}(\alpha) + SK\right)$$

for all $K \in \mathcal{K}_\alpha$.

Our second hypothesis replaces Assumption 3.2 as a technical condition that is used in the proof of Theorem 3.3.1 below.

Assumption 3.4. *Let $\tilde{\alpha}, \alpha \in \overline{\mathcal{A}}_\alpha$. Then there exists an $n \times n$ matrix*

$$J(\tilde{\alpha}, \alpha) = \sum_{i,j} \left(\tilde{\alpha}^{(i)} - \alpha^{(i)}\right)\left(\tilde{\alpha}^{(j)} - \alpha^{(j)}\right) J_{ij}(\alpha) \geq 0$$

such that

$$\begin{aligned} J(\tilde{\alpha}, \alpha) \leq{}& \left(T_{\theta_j}(\tilde{\alpha}) - T_{\theta_j}(\alpha)\right)' R \left(T_{\theta_j}(\tilde{\alpha}) - T_{\theta_j}(\alpha)\right) \\ &+ \left(T_{\theta_j}(\alpha) + SK\right)' J(\tilde{\alpha}, \alpha) \left(T_{\theta_j}(\alpha) + SK\right) \end{aligned} \tag{3.17}$$

for all $K \in \mathcal{K}_\alpha$ and $\theta_j \in \mathcal{V}$, and

$$\operatorname{tr}\left(J(\tilde{\alpha}, \alpha) Q\right) \geq \lambda \|\tilde{\alpha} - \alpha\|^2, \tag{3.18}$$

where $\lambda > 0$.

Our second result is closely related to Theorem 3.2.2. We show that least squares estimates for the true unknown parameter value α_0 remain strongly consistent in the distributed framework.

Theorem 3.3.1. *Under any control policy* $\{K_j, j = 0, 1, \dots\}$ *taking values in* $\mathcal{K}_\alpha$*, the sequence of least squares estimates* $\{\hat{\alpha}_k, k = 1, 2, \dots\}$*, obtained by minimizing the sum of squares* (3.16) *over* $\overline{\mathcal{A}}_\alpha$*, is strongly consistent for* $\alpha_0 \in \mathcal{A}_\alpha$*. That is,* $\hat{\alpha}_k \xrightarrow{\text{a.s.}} \alpha_0$ *as* $k \to \infty$*.*

Proof. First, observe that $\xi'_{j+1} R (H - H^0) X_j$ is a scalar, so it is equal to its transpose, $X'_j (H - H^0)' R \xi_{j+1}$. By the definition of the system (3.1), we have

$$\begin{aligned}
\mathcal{L}_k(\alpha) &= \sum_{j=0}^{k-1} \left(T^0_{\theta_j} X_j + S U_j + \xi_{j+1} - \left[T_{\theta_j} X_j + S U_j \right] \right)' \\
&\qquad \cdot R \left(T^0_{\theta_j} X_j + S U_j + \xi_{j+1} - \left[T_{\theta_j} X_j + S U_j \right] \right) \\
&= \sum_{j=0}^{k-1} \left(\left(1 - \tilde{t}\right) H^0_{\theta_j} X_j + \frac{\tilde{t}}{n} \mathbf{1} + S U_j + \xi_{j+1} - \left[\left(1 - \tilde{t}\right) H_{\theta_j} X_j + \frac{\tilde{t}}{n} \mathbf{1} + S U_j \right] \right)' \\
&\qquad \cdot R \left(\left(1 - \tilde{t}\right) H^0_{\theta_j} X_j + \frac{\tilde{t}}{n} \mathbf{1} + S U_j + \xi_{j+1} - \left[\left(1 - \tilde{t}\right) H_{\theta_j} X_j + \frac{\tilde{t}}{n} \mathbf{1} + S U_j \right] \right) \\
&= \sum_{j=0}^{k-1} \left(\xi_{j+1} - \left(1 - \tilde{t}\right) \left(H_{\theta_j} - H^0_{\theta_j} \right) X_j \right)' R \left(\xi_{j+1} - \left(1 - \tilde{t}\right) \left(H_{\theta_j} - H^0_{\theta_j} \right) X_j \right) \\
&= \sum_{j=0}^{k-1} \left(\xi'_{j+1} - \left(1 - \tilde{t}\right) X'_j \left(H_{\theta_j} - H^0_{\theta_j} \right)' \right) R \left(\xi_{j+1} - \left(1 - \tilde{t}\right) \left(H_{\theta_j} - H^0_{\theta_j} \right) X_j \right) \\
&= \sum_{j=0}^{k-1} \xi'_{j+1} R \xi_{j+1} + \left(1 - \tilde{t}\right)^2 \sum_{j=0}^{k-1} X'_j \left(H_{\theta_j} - H^0_{\theta_j} \right)' R \left(H_{\theta_j} - H^0_{\theta_j} \right) X_j \\
&\qquad - 2 \left(1 - \tilde{t}\right) \sum_{j=0}^{k-1} \xi'_{j+1} R \left(H_{\theta_j} - H^0_{\theta_j} \right) X_j .
\end{aligned}$$

Since

$$\sum_{j=0}^{k-1} \xi'_{j+1} R \xi_{j+1} = \mathcal{L}_k (\alpha_0) ,$$

we may write

$$\begin{aligned}
\mathcal{L}_k(\alpha) - \mathcal{L}_k (\alpha_0) &= \left(1 - \tilde{t}\right)^2 \sum_{j=0}^{k-1} X'_j \left(H_{\theta_j} - H^0_{\theta_j} \right)' R \left(H_{\theta_j} - H^0_{\theta_j} \right) X_j \\
&\qquad - 2 \left(1 - \tilde{t}\right) \sum_{j=0}^{k-1} \xi'_{j+1} R \left(H_{\theta_j} - H^0_{\theta_j} \right) X_j . \qquad (3.19)
\end{aligned}$$

Let the sequence $\{Y_j, j = 0, 1, \dots\}$ be defined by

$$Y_j = 2\left(1 - \tilde{t}\right) \xi'_{j+1} R\left(H_{\theta_j} - H^0_{\theta_j}\right) X_j, \quad j = 0, 1, \dots.$$

Then, since

$$\begin{aligned}\mathbb{E}\left[Y_j \mid \mathcal{F}_j\right] &= 2\left(1 - \tilde{t}\right) \mathbb{E}\left[\xi'_{j+1} \mid \mathcal{F}_j\right] R\left(H_{\theta_j} - H^0_{\theta_j}\right) X_j \\ &= 0, \quad \text{a.s.}, \ j = 0, 1, \dots,\end{aligned}$$

we see that the Y_j, $j = 0, 1, \dots$, are martingale differences. Furthermore,

$$\begin{aligned}\mathbb{E}\left[Y_j^2 \mid \mathcal{F}_j\right] &= 4\left(1 - \tilde{t}\right)^2 X'_j \left(H_{\theta_j} - H^0_{\theta_j}\right)' R\, \mathbb{E}\left[\xi_{j+1}\xi'_{j+1} \mid \mathcal{F}_j\right] R\left(H_{\theta_j} - H^0_{\theta_j}\right) X_j \\ &= 4\left(1 - \tilde{t}\right)^2 X'_j \left(H_{\theta_j} - H^0_{\theta_j}\right)' RQR\left(H_{\theta_j} - H^0_{\theta_j}\right) X_j, \quad \text{a.s.}, \ j = 0, 1, \dots.\end{aligned}$$

From Lemma 3.2.1, it follows that

$$\sum_{k=0}^{\infty} (k+1)^{-2}\, \mathbb{E} Y_k^2 < \infty.$$

Hence, by Lemma B.0.1, we have

$$\lim_{k \to \infty} k^{-1} \sum_{j=0}^{k-1} Y_j = 0, \quad \text{a.s.}$$

By definition of Y_j, the expression on the left-hand side above is a linear form in $\alpha - \alpha_0$ that converges to 0 almost surely, for arbitrary α, as $k \to \infty$. Indeed, since

$$\begin{aligned}Y_j &= 2\left(1 - \tilde{t}\right) \xi'_{j+1} R\left(H_{\theta_j} - H^0_{\theta_j}\right) X_j \\ &= 2\left(1 - \tilde{t}\right) \xi'_{j+1} R\left(\left[F^{(0)}_{\theta_j} + \alpha^{(1)} F^{(1)}_{\theta_j} + \cdots + \alpha^{(q)} F^{(q)}_{\theta_j}\right] - \left[F^{(0)}_{\theta_j} + \alpha^{(1)}_0 F^{(1)}_{\theta_j} + \cdots + \alpha^{(q)}_0 F^{(q)}_{\theta_j}\right]\right) X_j \\ &= \sum_{i=1}^{q} 2\left(1 - \tilde{t}\right) \left(\alpha^{(i)} - \alpha^{(i)}_0\right) \xi'_{j+1} R F^{(i)}_{\theta_j} X_j\end{aligned}$$

we obtain

$$\begin{aligned}\sum_{j=0}^{k-1} Y_j &= \sum_{j=0}^{k-1}\sum_{i=1}^{q} 2\left(1-\tilde{t}\right)\left(\alpha^{(i)}-\alpha_0^{(i)}\right)\xi_{j+1}' R F_{\theta_j}^{(i)} X_j \\ &= \sum_{i=1}^{q}\left(\alpha^{(i)}-\alpha_0^{(i)}\right)\left[\sum_{j=0}^{k-1} 2\left(1-\tilde{t}\right)\xi_{j+1}' R F_{\theta_j}^{(i)} X_j\right] \\ &= \sum_{i=1}^{q}\left(\alpha^{(i)}-\alpha_0^{(i)}\right)\ell_i(k),\end{aligned}$$

where

$$\ell_i(k) = \sum_{j=0}^{k-1} 2\left(1-\tilde{t}\right)\xi_{j+1}' R F_{\theta_j}^{(i)} X_j.$$

Hence,

$$k^{-1}\sum_{j=0}^{k-1} Y_j = k^{-1}\sum_{i=1}^{q}\left(\alpha^{(i)}-\alpha_0^{(i)}\right)\ell_i(k) \tag{3.20}$$

and it follows that

$$\lim_{k\to\infty} k^{-1}\|\ell(k)\| = 0, \quad \text{a.s.} \tag{3.21}$$

We next consider the first sum on the right-hand side of (3.8). Set, for $J = J(\alpha, \alpha_0)$,

$$Z_j = X_{j+1}' J X_{j+1} - X_j' J X_j + X_j'\left(T_{\theta_j} - T_{\theta_j}^0\right)' R\left(T_{\theta_j} - T_{\theta_j}^0\right) X_j - \operatorname{tr}(JQ),\ j = 0, 1, \ldots.$$

From (3.17), it follows that

$$\begin{aligned}\mathbb{E}\left[Z_j \mid \mathcal{F}_j\right] &= \mathbb{E}\left[X_{j+1}' J X_{j+1} \mid \mathcal{F}_j\right] - X_j' J X_j + X_j'\left(T_{\theta_j} - T_{\theta_j}^0\right)' R\left(T_{\theta_j} - T_{\theta_j}^0\right) X_j - \operatorname{tr}(JQ) \\ &= X_j'(T_{\theta_j} + SK)' J (T_{\theta_j} + SK) X_j + \mathbb{E}\left[\xi_{j+1}' J \xi_{j+1} \mid \mathcal{F}_j\right] \\ &\quad - X_j' J X_j + X_j'\left(T_{\theta_j} - T_{\theta_j}^0\right)' R\left(T_{\theta_j} - T_{\theta_j}^0\right) X_j - \operatorname{tr}(JQ) \\ &= X_j'(T_{\theta_j} + SK)' J (T_{\theta_j} + SK) X_j - X_j' J X_j + X_j'\left(T_{\theta_j} - T_{\theta_j}^0\right)' R\left(T_{\theta_j} - T_{\theta_j}^0\right) X_j \\ &\geq 0,\ j = 0, 1, \ldots.\end{aligned} \tag{3.22}$$

Next, we introduce the martingale

$$M_k = \sum_{j=0}^{k-1} \left(Z_j - \mathbb{E}\left[Z_j \mid \mathcal{F}_j \right] \right), \ k = 1, 2, \ldots .$$

From (3.22) and the fact that

$$\begin{aligned} T_{\theta_j} - T_{\theta_j}^0 &= \left[\left(1 - \tilde{t}\right) H_{\theta_j} + \frac{\tilde{t}}{n} \mathbf{1}\mathbf{1}' \right] - \left[\left(1 - \tilde{t}\right) H_{\theta_j}^0 + \frac{\tilde{t}}{n} \mathbf{1}\mathbf{1}' \right] \\ &= \left(1 - \tilde{t}\right) \left(H_{\theta_j} - H_{\theta_j}^0 \right) \end{aligned}$$

it follows that

$$\begin{aligned} M_k &\leq \sum_{j=0}^{k-1} Z_j \\ &= \sum_{j=0}^{k-1} \left[X_{j+1}' J X_{j+1} - X_j' J X_j + \left(1 - \tilde{t}\right)^2 X_j' \left(H_{\theta_j} - H_{\theta_j}^0 \right)' R \left(H_{\theta_j} - H_{\theta_j}^0 \right) X_j - \operatorname{tr}(JQ) \right] \\ &= X_k' J X_k - k \operatorname{tr}(JQ) + \left(1 - \tilde{t}\right)^2 \sum_{j=0}^{k-1} X_j' \left(H_{\theta_j} - H_{\theta_j}^0 \right)' R \left(H_{\theta_j} - H_{\theta_j}^0 \right) X_j, \ k = 1, 2, \ldots . \end{aligned}$$

Hence, by (3.7), we have

$$\begin{aligned} & k^{-1} M_k - k^{-1} X_k' J X_k + \lambda \|\alpha - \alpha_0\|^2 \\ & \qquad \leq k^{-1} \left(1 - \tilde{t}\right)^2 \sum_{j=0}^{k-1} X_j' \left(H_{\theta_j} - H_{\theta_j}^0 \right)' R \left(H_{\theta_j} - H_{\theta_j}^0 \right) X_j, \ k = 1, 2, \ldots . \end{aligned} \tag{3.23}$$

Lemma 3.2.1 implies that Lemma B.0.1 holds for $\{M_k, k = 1, 2, \ldots\}$ and that

$$\lim_{k \to \infty} k^{-1} X_k' J X_k = 0, \quad \text{a.s.}$$

The sum of the first two terms on the left-hand side of (3.23) is a quadratic form in $\alpha - \alpha_0$ that converges to 0 almost surely, for arbitrary α, as $k \to \infty$. Consequently, there exists an

almost surely finite random variable N with the property that

$$k^{-1}M_k - k^{-1}X_k'JX_k \geq -\frac{\lambda}{2}\|\alpha - \alpha_0\|^2 \tag{3.24}$$

for all α and for $k \geq N$. By (3.19) and (3.20), we have

$$\begin{aligned} k^{-1}\left(\mathcal{L}_k(\alpha) - \mathcal{L}_k(\alpha_0)\right) = k^{-1}\left(1-\tilde{t}\right)^2 \sum_{j=0}^{k-1} X_j' \left(H_{\theta_j} - H_{\theta_j}^0\right)' R \left(H_{\theta_j} - H_{\theta_j}^0\right) X_j \\ - k^{-1}\sum_{i=1}^{q}\left(\alpha^{(i)} - \alpha_0^{(i)}\right)\ell_i(k). \end{aligned}$$

Hence, it follows from (3.23) and (3.24) that

$$k^{-1}\left(\mathcal{L}_k(\alpha) - \mathcal{L}_k(\alpha_0)\right) \geq \frac{\lambda}{2}\|\alpha - \alpha_0\|^2 - k^{-1}\|\alpha - \alpha_0\|\,\|\ell(k)\|$$

for all α whenever $k \geq N$. Since $\mathcal{L}_k(\hat{\alpha}_k) = \min_{\alpha} \mathcal{L}_k(\alpha)$, we have

$$\mathcal{L}_k(\hat{\alpha}_k) - \mathcal{L}_k(\alpha_0) \leq 0,$$

and thus

$$\frac{\lambda}{2}\|\hat{\alpha}_k - \alpha_0\| \leq k^{-1}\|\ell(k)\| \tag{3.25}$$

for $k \geq N$. Hence, by (3.21), we conclude that

$$\lim_{k\to\infty} \|\hat{\alpha}_k - \alpha_0\| = 0, \quad \text{a.s.},$$

as desired. □

Fix $\alpha_0^* \in \mathcal{A}_\alpha$ and let $\{\alpha_j^*, j = 1, 2, \dots\}$ be defined by (3.15), assuming now that $\{\hat{\alpha}_j, j = 1, 2, \dots\}$ refers to the sequence of distributed (as opposed to centralized) least squares estimates. Since α_0 is contained in the open set $\mathcal{A}_\alpha$, it follows from Theorem 3.3.1 that, with probability one, $\hat{\alpha}_j$ must always lie within $\mathcal{A}_\alpha$ for j large enough. In particular, we have the

following corollary.

Corollary 3.3.2. *Under any control policy* $\{K_j, j = 0, 1, \dots\}$ *taking values in* $\mathcal{K}_\alpha$*, we have* $\lim_{j\to\infty} \alpha_j^* = \alpha_0$ a.s.

3.4 Simulations

In this section, we present numerical simulations of the least squares estimation of the unknown parameter α in the stochastic system model for PageRank. We shall consider the cases where α is one-, two-, and three-dimensional. Throughout this section, we take $R = I$ and employ a control policy $\{K_j, j = 0, 1, \dots\}$ that is identically zero. For each example, the web graph is randomly generated according to the procedure described in Section 2.3.

Example 3.2 (1D Centralized LSE)**.** Consider a web of $n = 200$ pages and 640 hyperlinks. We assume that the corresponding stochastic system model for PageRank depends on a one-dimensional unknown parameter α. Suppose first that the noise $\{\xi_j, j = 0, 1, \dots\}$ is identically zero. In this case, which is illustrated in Figure 3.1, the sequence of centralized least squares estimates converges immediately to the true value $\alpha_0 = 1/2$ and the sequence of PageRank iterates from the power method also convergences.

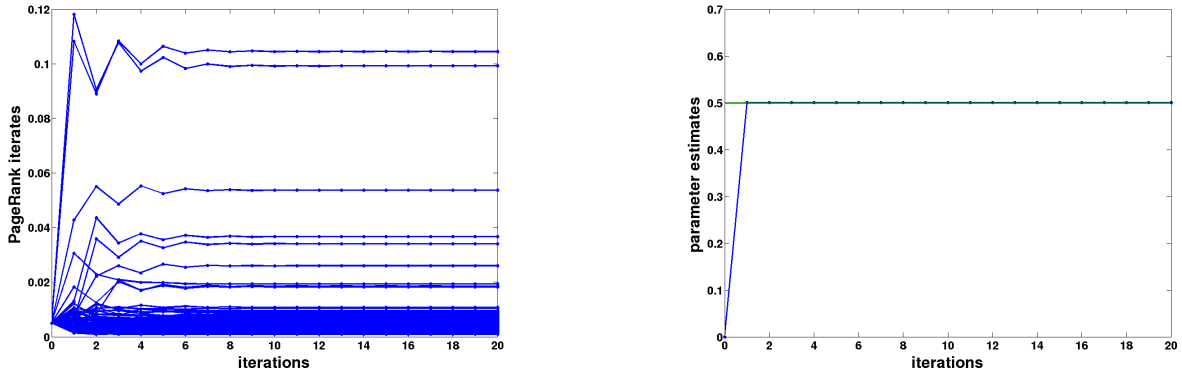


Figure 3.1: PageRank iterates and parameter estimates for Ex. 3.2 (zero noise)

Next, we introduce the noise terms, $\{\xi_j, j = 0, 1, \dots\}$, which are taken to be i.i.d. with zero mean and covariance matrix $Q = \sigma^2 I$, where $\sigma^2 = 10^{-5}$. In this case, the sequence of

centralized least squares estimates still converges to $\alpha_0 = 1/2$, although the PageRank iterates do not converge. This is seen in Figure 3.2.

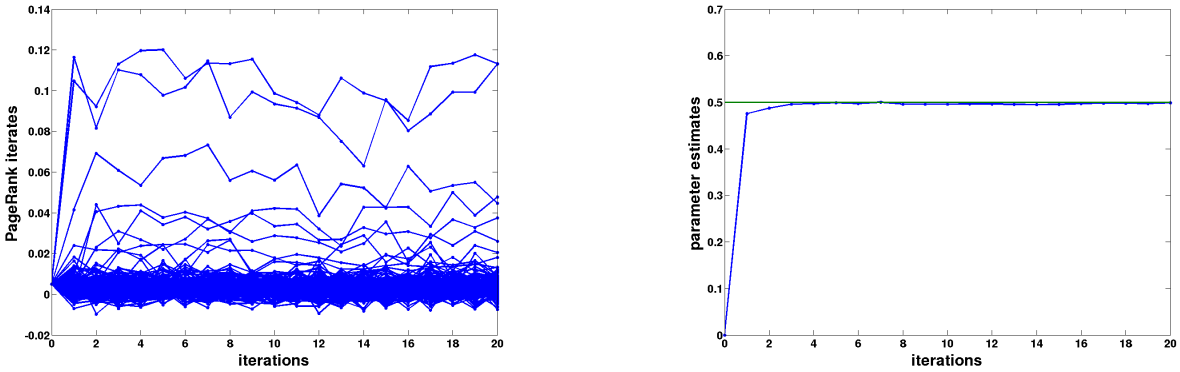


Figure 3.2: PageRank iterates and parameter estimates for Ex. 3.2 (nonzero noise)

Throughout the remaining examples of this chapter, we will continue to assume that the noise vectors $\{\xi_j, j = 0, 1, \dots\}$ have covariance matrix $Q = \sigma^2 I$, where $\sigma^2 = 10^{-5}$.

Example 3.3 (2D Centralized LSE)**.** Consider a web of $n = 200$ pages and 722 hyperlinks. We assume that the corresponding stochastic system model for PageRank depends on a two-dimensional unknown parameter α. Figure 3.3 illustrates the convergence of the sequence of centralized least squares parameter estimates to the true value $\alpha_0 = (1/2, -1)$.

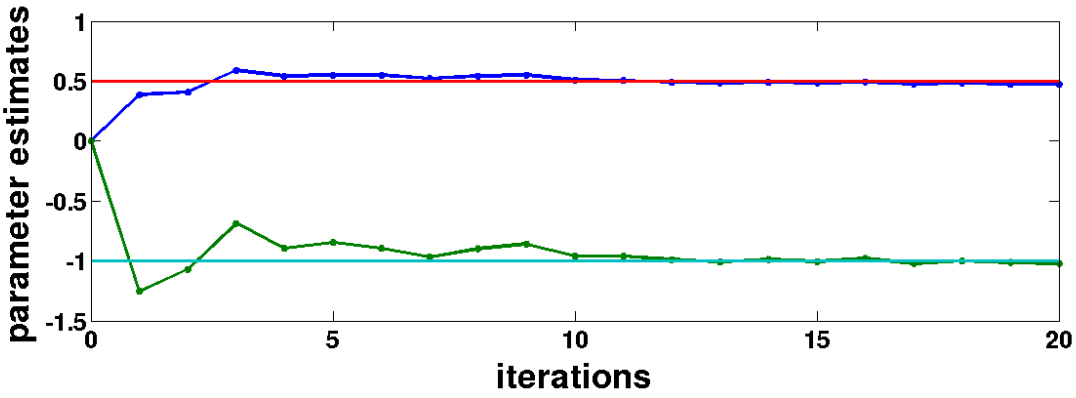


Figure 3.3: Centralized least squares parameter estimation for Ex. 3.3

Example 3.4 (3D Centralized LSE)**.** Consider a web of $n = 200$ pages and 609 hyperlinks. We assume that the corresponding stochastic system model for PageRank depends on a three-dimensional unknown parameter α. Figure 3.4 illustrates the convergence of the sequence of centralized least squares parameter estimates to the true value $\alpha_0 = (1/2, -1, 3/2)$.

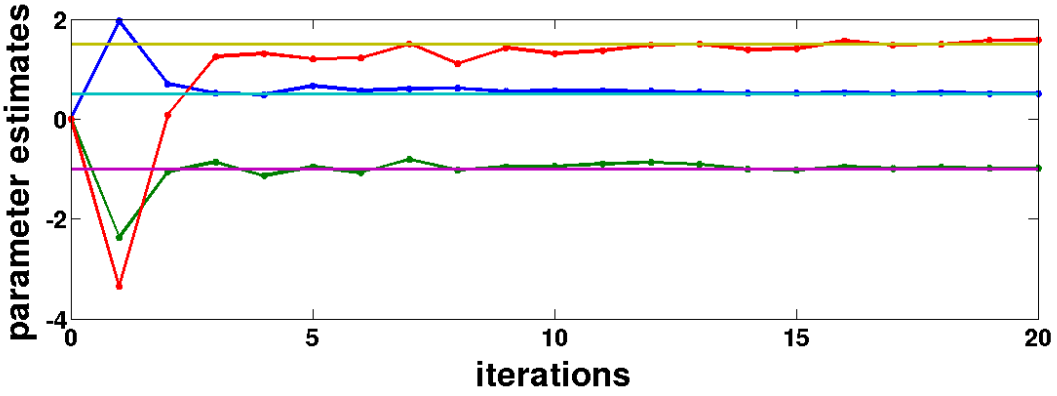


Figure 3.4: Centralized least squares parameter estimation for Ex. 3.4

Example 3.5 (1D Distributed LSE)**.** Consider a web of $n = 200$ pages and 881 hyperlinks. We assume that the corresponding stochastic system model for PageRank depends on a one-dimensional unknown parameter α. Figure 3.5 illustrates the convergence of the sequence of distributed least squares parameter estimates to the true value $\alpha_0 = 1/2$.

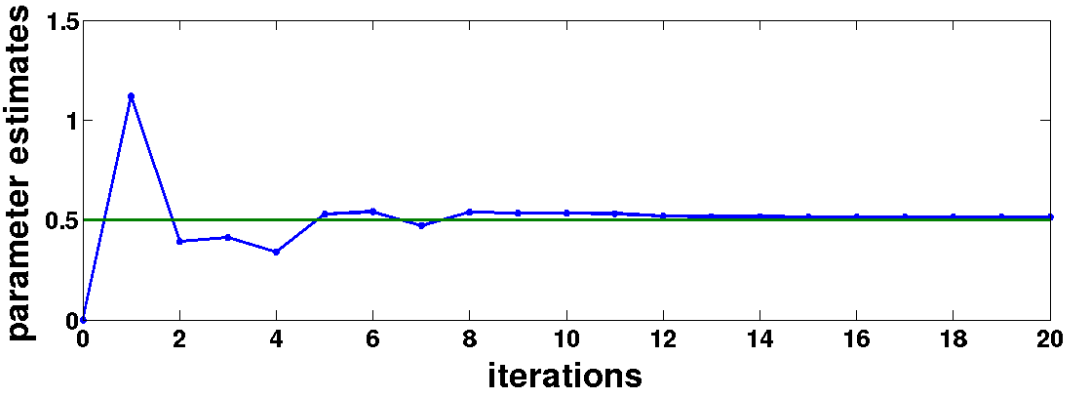


Figure 3.5: Distributed least squares parameter estimation for Ex. 3.5

Chapter 4

Adaptive Control of the Stochastic PageRank System

In Chapter 3, we presented a stochastic system model for PageRank and postulated the existence of an unknown parameter on which the dynamics of the system depend. The fact that the parameter is unknown implies that the system is unknown, so we proceeded to estimate the parameter via online least squares methods. We showed that the least squares estimator is strongly consistent, converging asymptotically to the true value of the parameter with probability one.

We now turn our attention to the adaptive control of the stochastic system model for PageRank. Using the online least squares estimates of the unknown parameter, we construct an adaptive control policy. For more on adaptive control, we refer to the texts of Åström, Goodwin, and Kumar [3], Åström and Murray [4], Åström and Wittenmark [5], Chen and Guo [13], Davis and Vinter [14], Kumar and Varaiya [33], as well as the monograph of Pasik-Duncan [47], the work of Duncan and Pasik-Duncan [15], and the references listed therein.

The remainder of this chapter is organized as follows. In Section 4.1, we recall the notion of stability from Chapter 3 and define what we mean by an optimal control policy. In

Section 4.2, we assume a minimum variance cost criterion, while in Section 4.3 we consider the related problem of adaptive tracking. In Section 4.4, we introduce a quadratic cost criterion and iteratively construct a control policy via dynamic programming techniques. In all three cases, we achieve a control policy whose performance according to the long-run average cost is equivalent to the optimal stationary control that would be used if we had knowledge of the true value of the parameter. Finally, in Section 4.5, we present numerical simulations that illustrate the convergence and optimality of the adaptive control policies.

4.1 Stability and Optimality

Recall that the process $\{X_j, j = 0, 1, \dots\}$ is *stable* under the control policy $\{K_j, j = 0, 1, \dots\}$, and that the control is also stable, if

$$\limsup_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} \mathbb{E}_x \|X_j\|^2 \le C, \tag{4.1}$$

where $\mathbb{E}_x$ denotes expectation conditional on the event $X_0 = x$ and C is a finite constant independent of x.

Based on this definition, we make the following observations.

Corollary 4.1.1. *If $\{X_j, j = 0, 1, \dots\}$ is stable, then*

$$\sum_{j=1}^{\infty} j^{-2}\, \mathbb{E}\|X_j\|^2 < \infty.$$

Corollary 4.1.2. *If $\{K_j, j = 0, 1, \dots\}$ is a stationary control policy satisfying $K_j = K$ for $j = 0, 1, \dots$, then $\{X_j, j = 0, 1, \dots\}$ is stable if and only if all the eigenvalues of $T + SK$ lie within the unit circle.*

We begin by considering a stationary control policy $K_j = K$, $j = 0, 1, \dots$. For a given cost criterion $\{C_k, k = 0, 1, \dots\}$, we shall say that the stationary control K is *optimal* if it is

stable and if

$$\lim_{k\to\infty} k^{-1}C_k = \gamma \quad \text{a.s. under } K, \tag{4.2}$$

where γ is a constant such that

$$\lim_{k\to\infty} k^{-1}C_k \geq \gamma \quad \text{a.s.} \tag{4.3}$$

under an arbitrary stable control policy $\{K_j, j = 0, 1, \dots\}$.

We shall consider various cost criteria and, in each case, our control objective will be to minimize the ergodic cost $\lim_{k\to\infty} k^{-1}C_k$.

4.2 Minimum Variance Control

We begin by considering a minimum variance control scheme. Although this type of control is not directly applicable to PageRank, since its effect is to drive the state of the system to zero, it is a useful starting point from which to develop a foundation that will be used in Section 4.3 on the adaptive tracking of a specified reference trajectory. For an approachable introduction to the general strategy of constructing a minimum variance controller, we refer to the text of Söderström [50, p. 297].

Let $\gamma = \operatorname{tr}(Q)$ and observe that

$$\begin{aligned}
\mathbb{E}\left[\|X_j\|^2 \mid \mathcal{F}_{j-1}\right] &= \mathbb{E}\left[X_j' X_j \mid \mathcal{F}_{j-1}\right] \\
&= \mathbb{E}\left[(TX_{j-1} + SU_{j-1} + \xi_j)'(TX_{j-1} + SU_{j-1} + \xi_j) \mid \mathcal{F}_{j-1}\right] \\
&= \mathbb{E}\left[\left((TX_{j-1} + SU_{j-1})' + \xi_j'\right)((TX_{j-1} + SU_{j-1}) + \xi_j) \mid \mathcal{F}_{j-1}\right] \\
&= \mathbb{E}\left[(TX_{j-1} + SU_{j-1})'(TX_{j-1} + SU_{j-1})' \mid \mathcal{F}_{j-1}\right] + \mathbb{E}\left[\xi_j'\xi_j \mid \mathcal{F}_{j-1}\right] \\
&\quad + \mathbb{E}\left[(TX_{j-1} + SU_{j-1})'\xi_j \mid \mathcal{F}_{j-1}\right] + \mathbb{E}\left[\xi_j'(TX_{j-1} + SU_{j-1}) \mid \mathcal{F}_{j-1}\right] \\
&= (TX_{j-1} + SU_{j-1})'(TX_{j-1} + SU_{j-1}) + \gamma.
\end{aligned} \tag{4.4}$$

Thus, in order to minimize $\mathbb{E}\left[\|X_j\|^2\right] = \mathbb{E}\left[\mathbb{E}\left[\|X_j\|^2 \mid \mathcal{F}_{j-1}\right]\right]$, we must take

$$TX_{j-1} + SU_{j-1} = 0.$$

For the feedback control $U_{j-1} = KX_{j-1}$, it follows that

$$(T + SK)X_{j-1} = 0,$$

so we seek K such that

$$SK = -T.$$

Provided that the $n \times m$ matrix S has full rank $m \leq n$, there exists a (not necessarily unique) left-inverse, $S_L^{-1} \in \mathbb{R}^{m \times n}$, defined by

$$S_L^{-1} = (S'S)^{-1}S',$$

with the property that $S_L^{-1}S = I$. Thus, the control $K^{\mathrm{MV}} = -S_L^{-1}T$ minimizes $\mathbb{E}\left[\|X_j\|^2\right]$. In the case that $T = T(\alpha)$ depends on the unknown parameter α, we take

$$K^{\mathrm{MV}}(\alpha) = -S_L^{-1}T(\alpha).$$

We now consider a control policy that is constructed as follows. At time 0, we select $\alpha_0^* \in \mathcal{A}_\alpha$ and set $K_0 = K^{\mathrm{MV}}\left(\alpha_0^*\right)$. For $j = 1, 2, \ldots$, we observe $\{X_0, \ldots, X_j\}$, let the estimate $\hat{\alpha}_j$ be the minimizer of the least squares functional (3.3) (or (3.16)), and set

$$K_j^{\mathrm{MV}} = K^{\mathrm{MV}}\left(\alpha_j^*\right),$$

where α_j^* is defined by (3.15). The matrices K_j^{MV}, $j = 0, 1, \ldots$, are estimates of the optimal stationary control, $K^0 = K^{\mathrm{MV}}(\alpha_0)$.

By the affine dependence of $T(\alpha)$ on α, it follows that $K^{\mathrm{MV}}(\alpha)$ is differentiable arbitrarily-many times with respect to $\alpha \in \mathcal{A}_\alpha$. Thus, we may expand $K^{\mathrm{MV}}(\alpha)$ as a Taylor series about α_0:

$$K^{\mathrm{MV}}(\alpha) = K^{\mathrm{MV}}(\alpha_0) + \sum_{i=1}^{q} \left(\alpha^{(i)} - \alpha_0^{(i)}\right) \frac{\partial}{\partial \alpha^{(i)}} K^{\mathrm{MV}}(\alpha_0) + O\left(\|\alpha - \alpha_0\|^2\right).$$

Hence, by Corollary 3.2.3 (or Corollary 3.3.2, respectively), it follows that

$$K_j^{\mathrm{MV}} = K^{\mathrm{MV}}\left(\alpha_j^*\right) \xrightarrow{\text{a.s.}} K^{\mathrm{MV}}(\alpha_0) = K^0 \text{ as } j \to \infty. \tag{4.5}$$

Now let

$$C_k^{\mathrm{MV}} = \sum_{j=1}^{k} \mathbb{E}\left[\|X_j\|^2\right], \ k = 1, 2, \ldots,$$

be the minimum variance cost criterion and suppose that our control objective is to minimize

$$\lim_{k\to\infty} k^{-1} C_k^{\mathrm{MV}}.$$

Under the control policy $\{K_j^{\mathrm{MV}}, j = 0, 1, \ldots\}$ constructed above, we have $SK^0 = -T^0$, and it follows that

$$\begin{aligned}
\left(T^0 X_{j-1} + SU_{j-1}\right)' \left(T^0 X_{j-1} + SU_{j-1}\right) &= \left(-SK^0 X_{j-1} + SK_{j-1}^{\mathrm{MV}} X_{j-1}\right)' \left(-SK^0 X_{j-1} + SK_{j-1}^{\mathrm{MV}} X_{j-1}\right) \\
&= \left(S\left(K_{j-1}^{\mathrm{MV}} - K^0\right) X_{j-1}\right)' \left(S\left(K_{j-1}^{\mathrm{MV}} - K^0\right) X_{j-1}\right) \\
&= \left\|S\left(K_{j-1}^{\mathrm{MV}} - K^0\right) X_{j-1}\right\|^2.
\end{aligned} \tag{4.6}$$

Lemma 4.2.1. *If $\{K_j, j = 0, 1, \ldots\}$ is a stable control policy for which $K_j \xrightarrow{\text{a.s.}} K^0$, then*

$$\limsup_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} \|X_j\|^4 < \infty, \text{ a.s.} \tag{4.7}$$

The proof relies on the fact that K^0 is a stable stationary control. Indeed, under the

policy $\{K_j = K^0, j = 0, 1, \dots\}$, we have

$$\begin{aligned}
\mathbb{E}\left[\|X_j\|^2 \mid \mathcal{F}_{j-1}\right] &= \left(T^0 X_{j-1} + SU_{j-1}\right)' \left(T^0 X_{j-1} + SU_{j-1}\right) + \gamma \\
&= \left(-SK^0 X_{j-1} + SK^0 X_{j-1}\right)' \left(-SK^0 X_{j-1} + SK^0 X_{j-1}\right) + \gamma \\
&= \gamma.
\end{aligned}$$

Hence,

$$\begin{aligned}
\limsup_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} \mathbb{E}\left[\|X_j\|^2\right] &= \limsup_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} \mathbb{E}\left[\mathbb{E}\left[\|X_j\|^2 \mid \mathcal{F}_j\right]\right] \\
&= \limsup_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} \mathbb{E}\left[\gamma\right] \\
&= \limsup_{k\to\infty} k^{-1}(k\gamma) \\
&= \gamma < \infty.
\end{aligned}$$

Since K^0 is stable, all eigenvalues of $T^0 + SK^0$ lie within the unit circle. To establish (4.7), it suffices to show that

$$\limsup_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} \|X_j\|^4 < \infty$$

holds outside a set of arbitrarily small measure. See [43] for complete details.

Lemma 4.2.2. *Suppose that the cost criterion $\{C_k, k = 1, 2, \dots\}$ satisfies*

$$C_k = \sum_{j=1}^{k} \mathbb{E}\left[\left\|S\left(K_{j-1} - K^0\right) X_{j-1}\right\|^2 + \gamma\right], \tag{4.8}$$

where

$$\left\|K_j - K^0\right\| \xrightarrow{\text{a.s.}} 0 \text{ as } j \to \infty. \tag{4.9}$$

Then it follows that

$$\lim_{k\to\infty} k^{-1} C_k \le \gamma, \text{ a.s.} \tag{4.10}$$

under the control policy $\{K_j, j = 0, 1, \dots\}$.

Proof. By Fatou's lemma and the Cauchy-Schwarz inequality, we have

$$\begin{aligned}
\lim_{k\to\infty} k^{-1} C_k &= \lim_{k\to\infty} k^{-1} \sum_{j=1}^{k} \mathbb{E}\left[\left\|S\left(K_{j-1} - K^0\right) X_{j-1}\right\|^2 + \gamma\right] \\
&\le \gamma + \lim_{k\to\infty} k^{-1} \sum_{j=1}^{k} \mathbb{E}\left[\|S\|^2 \left\|K_{j-1} - K^0\right\|^2 \|X_{j-1}\|^2\right] \\
&\le \gamma + \mathbb{E}\left[\limsup_{k\to\infty} k^{-1}\|S\|^2 \sum_{j=1}^{k} \left\|K_{j-1} - K^0\right\|^2 \|X_{j-1}\|^2\right] \\
&\le \gamma + \mathbb{E}\left[\limsup_{k\to\infty} \|S\|^2 \left(k^{-1} \sum_{j=1}^{k} \left\|K_{j-1} - K^0\right\|^4\right)^{1/2} \left(k^{-1} \sum_{j=1}^{k} \|X_{j-1}\|^4\right)^{1/2}\right].
\end{aligned}$$

By (4.9) and Lemma 4.2.1, it follows that (4.10) holds under the policy $\{K_j, j = 0, 1, \dots\}$. □

We are now prepared to establish that the policy $\left\{K_j^{\mathrm{MV}}, j = 0, 1, \dots\right\}$ is not only stable but also equivalent to the optimal stationary control, K^0, with regards to the long term average of the minimum variance cost criterion.

Theorem 4.2.3. *Under the control policy* $\left\{K_j^{\mathrm{MV}}, j = 0, 1, \dots\right\}$*, we have*

$$\lim_{k\to\infty} k^{-1} C_k^{\mathrm{MV}} = \gamma, \text{ a.s.} \tag{4.11}$$

Proof. For the control policy $\left\{K_j^{\mathrm{MV}}, j = 0, 1, \dots\right\}$ and the corresponding minimum variance cost criterion $\left\{C_k^{\mathrm{MV}}, k = 0, 1, \dots\right\}$, the hypothesis of Lemma 4.2.2 is satisfied due to (4.4), (4.5), and (4.6). Thus, it follows from the lemma that $\lim_{k\to\infty} k^{-1} C_k^{\mathrm{MV}} \le \gamma$ a.s. under $\left\{K_j^{\mathrm{MV}}, j = 0, 1, \dots\right\}$. On the other hand, under any stable control policy, we have

$$\begin{aligned}
\limsup_{k\to\infty} k^{-1} C_k^{\mathrm{MV}} &= \limsup_{k\to\infty} k^{-1} \sum_{j=1}^{k} \mathbb{E}\left[\left(TX_{j-1} + SU_{j-1}\right)' \left(TX_{j-1} + SU_{j-1}\right) + \gamma\right] \\
&\ge \limsup_{k\to\infty} k^{-1} \sum_{j=1}^{k} \gamma \\
&= \gamma, \text{ a.s.}
\end{aligned}$$

Hence, we conclude that (4.11) holds under $\left\{K_j^{\mathrm{MV}}, j = 0, 1, \dots\right\}$. □

4.3 Adaptive Tracking

Suppose now that our goal in controlling the system is to track a specified reference trajectory, rather than to keep the output close to zero. This scenario has a more practical interpretation in the context of PageRank; namely, that we may wish to drive the PageRank vector approximations, represented by the sequence of state vectors of the stochastic system, toward a particular distribution across the pages in the web.

Let $\{Y_j, j = 0, 1, \dots\}$ be a given reference signal that is independent of $\{\mathcal{F}_j, j = 0, 1, \dots\}$ and observe that

$$\begin{aligned}
\mathbb{E}\left[\|X_j - Y_j\|^2 \mid \mathcal{F}_{j-1}\right] &= \mathbb{E}\left[(X_j - Y_j)'(X_j - Y_j) \mid \mathcal{F}_{j-1}\right] \\
&= \mathbb{E}\left[(TX_{j-1} + SU_{j-1} + \xi_j - Y_j)'(TX_{j-1} + SU_{j-1} + \xi_j - Y_j) \mid \mathcal{F}_{j-1}\right] \\
&= \mathbb{E}\left[\left((TX_{j-1} + SU_{j-1} - Y_j)' + \xi_j'\right)\left((TX_{j-1} + SU_{j-1} - Y_j) + \xi_j\right) \mid \mathcal{F}_{j-1}\right] \\
&= \mathbb{E}\left[(TX_{j-1} + SU_{j-1} - Y_j)'(TX_{j-1} + SU_{j-1} - Y_j)' \mid \mathcal{F}_{j-1}\right] + \mathbb{E}\left[\xi_j'\xi_j \mid \mathcal{F}_{j-1}\right] \\
&\quad + \mathbb{E}\left[(TX_{j-1} + SU_{j-1} - Y_j)'\xi_j \mid \mathcal{F}_{j-1}\right] + \mathbb{E}\left[\xi_j'(TX_{j-1} + SU_{j-1} - Y_j) \mid \mathcal{F}_{j-1}\right] \\
&= (TX_{j-1} + SU_{j-1} - Y_j)'(TX_{j-1} + SU_{j-1} - Y_j) + \gamma. \qquad (4.12)
\end{aligned}$$

Thus, in order to minimize $\mathbb{E}[\|X_j - Y_j\|^2] = \mathbb{E}[\mathbb{E}[\|X_j - Y_j\|^2 \mid \mathcal{F}_{j-1}]]$, we require

$$TX_{j-1} + SU_{j-1} - Y_j = 0.$$

For the feedback control $U_{j-1} = KX_{j-1}$, we have

$$(T + SK)X_{j-1} = Y_j.$$

In the case that $T = T(\alpha)$ depends on the unknown parameter α, we select $K^{\mathrm{AT}}(\alpha)$ such

that

$$\left(T(\alpha) + SK^{\mathrm{AT}}(\alpha)\right) X_{j-1} = Y_j.$$

We now consider a control policy that is constructed as follows. At time 0, we select $\alpha_0^* \in \mathcal{A}_\alpha$ and set $K_0 = K^{\mathrm{AT}}(\alpha_0^*)$. For $j = 1, 2, \ldots$, we observe $\{X_0, \ldots, X_j\}$, let the estimate $\hat{\alpha}_j$ be the minimizer of the least squares functional (3.3) (or (3.16)), and set

$$K_j^{\mathrm{AT}} = K^{\mathrm{AT}}\left(\alpha_j^*\right),$$

where α_j^* is defined by (3.15). The matrices K_j^{AT}, $j = 0, 1, \ldots$, are estimates of the optimal stationary control, $K^0 = K^{\mathrm{AT}}(\alpha_0)$.

As in the Section 4.2, it follows from Corollary 3.2.3 (or Corollary 3.2.3, respectively) and the affine dependence of $T(\alpha)$ on α that

$$K_j^{\mathrm{AT}} = K^{\mathrm{AT}}\left(\alpha_j^*\right) \xrightarrow{\text{a.s.}} K^{\mathrm{AT}}(\alpha_0) = K^0 \text{ as } j \to \infty. \tag{4.13}$$

Now let

$$C_k^{\mathrm{AT}} = \sum_{j=1}^{k} \mathbb{E}\left[\|X_j - Y_j\|^2\right], \ k = 1, 2, \ldots,$$

be the adaptive tracking cost criterion and suppose that our control objective is to minimize

$$\lim_{k\to\infty} k^{-1} C_k^{\mathrm{AT}}.$$

Under the control policy $\{K_j^{\mathrm{AT}}, j = 0, 1, \ldots\}$ constructed above, we have

$$\left(T^0 + SK^0\right) X_{j-1} = Y_j$$

and it follows that

$$\begin{aligned}\left(T^0X_{j-1}+SU_{j-1}-Y_j\right)'\left(T^0X_{j-1}+SU_{j-1}-Y_j\right) &= \left(T^0X_{j-1}+SK_{j-1}^{\mathrm{AT}}X_{j-1}-\left(T^0+SK^0\right)X_{j-1}\right)' \\ &\quad\cdot\left(T^0X_{j-1}+SK_{j-1}^{\mathrm{AT}}X_{j-1}-\left(T^0+SK^0\right)X_{j-1}\right) \\ &= \left(S\left(K_{j-1}^{\mathrm{AT}}-K^0\right)X_{j-1}\right)'\left(S\left(K_{j-1}^{\mathrm{AT}}-K^0\right)X_{j-1}\right) \\ &= \left\|S\left(K_{j-1}^{\mathrm{AT}}-K^0\right)X_{j-1}\right\|^2. \end{aligned} \tag{4.14}$$

We now establish that the policy $\left\{K_j^{\mathrm{AT}}, j=0,1,\dots\right\}$ is not only stable but also equivalent to the optimal stationary control, K^0, with regards to the long term average of the adaptive tracking cost criterion.

Theorem 4.3.1. *Under the control policy* $\left\{K_j^{\mathrm{AT}}, j=0,1,\dots\right\}$*, we have*

$$\lim_{k\to\infty} k^{-1}C_k^{\mathrm{AT}} = \gamma, \text{ a.s.} \tag{4.15}$$

Proof. For the control policy $\left\{K_j^{\mathrm{AT}}, j=0,1,\dots\right\}$ and the corresponding adaptive tracking cost criterion $\left\{C_k^{\mathrm{AT}}, k=0,1,\dots\right\}$, the hypothesis of Lemma 4.2.2 is satisfied due to (4.12), (4.13), and (4.14). Thus, the lemma implies that $\lim_{k\to\infty} k^{-1}C_k^{\mathrm{AT}} \le \gamma$ a.s. under $\left\{K_j^{\mathrm{AT}}, j=0,1,\dots\right\}$. On the other hand, under any stable control policy, we have

$$\begin{aligned}\limsup_{k\to\infty} k^{-1}C_k^{\mathrm{AT}} &= \limsup_{k\to\infty} k^{-1}\sum_{j=1}^{k}\mathbb{E}\left[\left(TX_{j-1}+SU_{j-1}-Y_j\right)'\left(TX_{j-1}+SU_{j-1}-Y_j\right)+\gamma\right] \\ &\ge \limsup_{k\to\infty} k^{-1}\sum_{j=1}^{k}\gamma \\ &= \gamma, \text{ a.s.}\end{aligned}$$

Hence, we conclude that (4.15) holds under $\left\{K_j^{\mathrm{AT}}, j=0,1,\dots\right\}$. □

4.4 Quadratic Cost Criterion

Motivated again by the work of Mandl [43], we finally consider the quadratic cost criterion

$$\begin{aligned} C_k^{\mathrm{LQ}} &= \sum_{j=0}^{k-1} \left(X_j' A X_j + U_j' B U_j \right) \\ &= \sum_{j=0}^{k-1} \left(X_j' A X_j + (K_j X_j)' B (K_j X_j) \right) \\ &= \sum_{j=0}^{k-1} X_j' \left(A + K_j' B K_j \right) X_j, \end{aligned}$$

where A is a non-negative definite $n \times n$ matrix and B is a positive definite $m \times m$ matrix.

Our first objective is to find an optimal stationary control, under the assumption that T and S are known, using a version of Howard's policy improvement algorithm of dynamic programming [21, p. 60]. Specifically, we wish to find a symmetric $n \times n$ matrix $W(T)$ such that

$$x'W(T)x = \min_u \left\{ x'Ax + u'Bu + (Tx + Su)'W(T)(Tx + Su) \right\}. \tag{4.16}$$

Let $u = K(T)x$ denote the value of u that minimizes the expression on the right-hand side of the above equation. It will be later shown that $K(T)$ is the optimal stationary control.

The following assumptions guarantee the stability of the system under the controls constructed through the iterative procedure of the policy improvement algorithm.

Assumption 4.1. *There exists a positive integer ℓ such that* $\sum_{j=0}^{\ell-1} \left(T^j \right)' A T^j > 0$.

Corollary 4.4.1. *If Assumption 4.1 holds, then*

$$D := \sum_{j=0}^{\ell-1} \left((T + SK)^j \right)' (A + K'BK) (T + SK)^j > 0$$

for an arbitrary $n \times m$ matrix K.

Besides Assumption 4.1, we make the following hypothesis.

Assumption 4.2. *There exists an $m \times n$ matrix $K^{(0)}$ such that all eigenvalues of $T + SK^{(0)}$ lie within the unit circle.*

Whenever K is a stable stationary control, let

$$W(K \mid T) = \sum_{k=0}^{\infty} \left((T + SK)^k\right)' (A + K'BK) (T + SK)^k.$$

Then $W = W(K \mid T)$ is the only symmetric matrix that satisfies

$$W = A + K'BK + (T + SK)'W(T + SK). \tag{4.17}$$

Indeed, since

$$\left((T + SK)^0\right)' (A + K'BK) (T + SK)^0 = A + K'SK$$

and

$$(T + SK)' \left[\sum_{k=0}^{\infty} \left((T + SK)^k\right)' (A + K'BK) (T + SK)^k\right] (T + SK) = \sum_{k=1}^{\infty} \left((T + SK)^k\right)' (A + K'BK) (T + SK)^k,$$

we see that $W = W(K \mid T)$ satisfies (4.17). Conversely, suppose that $\tilde{W}$ is any symmetric matrix satisfying (4.17). First, observe that

$$\begin{aligned}
\tilde{W} &= A + K'BK + (T + SK)'\tilde{W}(T + SK) \\
&= A + K'BK + (T + SK)' \left[A + K'BK + (T + SK)'\tilde{W}(T + SK)\right] (T + SK) \\
&= A + K'BK + (T + SK)' (A + K'BK) (T + SK) + \left((T + SK)^2\right)' \tilde{W}(T + SK)^2 \\
&= \sum_{k=0}^{1} \left((T + SK)^k\right)' (A + K'BK) (T + SK)^k + \left((T + SK)^2\right)' \tilde{W}(T + SK)^2.
\end{aligned}$$

If r is any positive integer such that

$$\tilde{W} = \sum_{k=0}^{r-1} \left((T+SK)^k\right)' (A+K'BK)(T+SK)^k + \left((T+SK)^r\right)' \tilde{W}(T+SK)^r,$$

then it follows that

$$\begin{aligned}
\tilde{W} &= \sum_{k=0}^{r-1} \left((T+SK)^k\right)' (A+K'BK)(T+SK)^k \\
&\qquad + \left((T+SK)^r\right)' \left[A+K'BK+(T+SK)'\tilde{W}(T+SK)\right](T+SK)^r \\
&= \sum_{k=0}^{r-1} \left((T+SK)^k\right)' (A+K'BK)(T+SK)^k \\
&\qquad + \left((T+SK)^r\right)' (A+K'BK)(T+SK)^r + \left((T+SK)^{r+1}\right)' \tilde{W}(T+SK)^{r+1} \\
&= \sum_{k=0}^{r} \left((T+SK)^k\right)' (A+K'BK)(T+SK)^k + \left((T+SK)^{r+1}\right)' \tilde{W}(T+SK)^{r+1}.
\end{aligned}$$

Hence, by induction on r, we must have $\tilde{W} = W(K \mid T)$.

The solution of (4.16) is obtained iteratively as follows. Starting with a matrix $K^{(0)}$ that satisfies Assumption 4.2, we construct a sequence $\{K^{(0)}, K^{(1)}, \dots\}$ such that

$$\begin{aligned}
&\left(K^{(i+1)}\right)' BK^{(i+1)} + \left(T+SK^{(i+1)}\right)' W\left(K^{(i)} \mid T\right)\left(T+SK^{(i+1)}\right) \\
&\qquad = \min_K \left\{K'BK + (T+SK)' W\left(K^{(i)} \mid T\right)(T+SK)\right\}.
\end{aligned} \tag{4.18}$$

This relation holds if and only if

$$K^{(i+1)} = -\left(B + S' W\left(K^{(i)} \mid T\right) S\right)^{-1} S' W\left(K^{(i)} \mid T\right) T. \tag{4.19}$$

Lemma 4.4.2. *If $\{X_j, j = 0, 1, \dots\}$ is stable under $K^{(i)}$, then it is also stable under $K^{(i+1)}$. Furthermore,*

$$W\left(K^{(i+1)} \mid T\right) \le W\left(K^{(i)} \mid T\right), \quad i = 0, 1, \dots \tag{4.20}$$

Proof. We begin by observing that the process $\{X_j, j = 0, 1, \dots\}$ is stable under $K^{(0)}$, which

is chosen to satisfy assumption 4.1. Now consider $\{X_j, j = 0, 1, \dots\}$ under the stationary control $K^{(m+1)}$). For convenience, we let

$$W = W\left(K^{(m)} \mid T\right).$$

Note that

$$X_0'\left(T + SK^{(m+1)}\right)' W\left(T + SK^{(m+1)}\right) X_0 = \mathbb{E}_x\left[X_1' W X_1\right] - \operatorname{tr}(WQ).$$

From (4.17) and (4.18), we have

$$W \geq A + \left(K^{(m+1)}\right)' B K^{(m+1)} + \left(T + SK^{(m+1)}\right)' W\left(T + SK^{(m+1)}\right) \tag{4.21}$$

and hence,

$$X_0' W X_0 \geq X_0'\left(A + \left(K^{(m+1)}\right)' B K^{(m+1)}\right) X_0 + \mathbb{E}_x\left[X_1' W X_1\right] - \operatorname{tr}(WQ).$$

In general,

$$\mathbb{E}_x\left[X_j' W X_j\right] \geq \mathbb{E}_x\left[X_j'\left(A + \left(K^{(m+1)}\right)' B K^{(m+1)}\right) X_j\right] + \mathbb{E}_x\left[X_{j+1}' W X_{j+1}\right] - \operatorname{tr}(WQ),\ j = 0, 1, \dots.$$

Let ℓ be the positive integer from Assumption 4.1. Then, summing the above inequality over $j = i, i+1, \dots, i+\ell k-1$ yields

$$\begin{aligned}\mathbb{E}_x\left[X_i' W X_i\right] + \ell k \operatorname{tr}(WQ) &\geq \sum_{j=0}^{\ell k-1} \mathbb{E}_x\left[X_{i+j}'\left(A + \left(K^{(m+1)}\right)' B K^{(m+1)}\right) X_{i+j}\right] \\ &\geq \sum_{p=0}^{k-1} \mathbb{E}_x\left[X_{i+p\ell}' D X_{i+p\ell}\right],\ i = 0, 1, \dots,\end{aligned}$$

where

$$D = \sum_{j=0}^{\ell-1} \left(\left(T + SK^{(m+1)} \right)^j \right)' \left(A + \left(K^{(m+1)} \right)' BK^{(m+1)} \right) \left(T + SK^{(m+1)} \right)^j .$$

By Corollary 4.4.1, we have $D > 0$. Hence, it follows that

$$\mathbb{E}_x \left[X_i' W X_i \right] + \ell k \operatorname{tr}(WQ) \geq C \sum_{j=0}^{k-1} \mathbb{E}_x \| X_{i+j\ell} \|^2.$$

Summing over $i = 0, 1, \ldots, \ell - 1$, we now have

$$\sum_{i=0}^{\ell-1} \mathbb{E}_x \left[X_i' W X_i \right] + \ell^2 k \operatorname{tr}(WQ) \geq C \sum_{j=0}^{\ell k-1} \| X_j \|^2,$$

which implies that

$$\limsup_{k \to \infty} k^{-1} \sum_{j=0}^{k-1} \mathbb{E}_x \| X_j \|^2 \leq \ell \operatorname{tr}(WQ)/C < \infty.$$

Hence, $\{X_j, j = 0, 1, \ldots\}$ is stable under the control $K^{(m+1)}$. To establish (4.20), we first conclude from (4.21) that

$$\begin{aligned}
&\left(\left(T + SK^{(m+1)} \right)^j \right)' W \left(T + SK^{(m+1)} \right)^j \\
&\quad \geq \left(\left(T + SK^{(m+1)} \right)^j \right)' \left(A + \left(K^{(m+1)} \right)' BK^{(m+1)} \right) \left(T + SK^{(m+1)} \right)^j \\
&\qquad + \left(\left(T + SK^{(m+1)} \right)^{j+1} \right)' W \left(T + SK^{(m+1)} \right)^{j+1}, \ j = 0, 1, \ldots .
\end{aligned}$$

Summing these inequalities yields

$$\begin{aligned}
&\sum_{j=0}^{\infty}\left(\left(T+SK^{(m+1)}\right)^{j}\right)'\left(A+\left(K^{(m+1)}\right)'BK^{(m+1)}\right)\left(T+SK^{(m+1)}\right)^{j}\\
&\quad\le\sum_{j=0}^{\infty}\left[\left(\left(T+SK^{(m+1)}\right)^{j}\right)'W\left(T+SK^{(m+1)}\right)^{j}-\left(\left(T+SK^{(m+1)}\right)^{j+1}\right)'W\left(T+SK^{(m+1)}\right)^{j+1}\right]\\
&\quad=W-\left(T+SK^{(m+1)}\right)'W\left(T+SK^{(m+1)}\right)+\left(T+SK^{(m+1)}\right)'W\left(T+SK^{(m+1)}\right)\\
&\qquad-\left(\left(T+SK^{(m+1)}\right)^{2}\right)'W\left(T+SK^{(m+1)}\right)^{2}+\left(\left(T+SK^{(m+1)}\right)^{2}\right)'W\left(T+SK^{(m+1)}\right)^{2}\\
&\qquad-\left(\left(T+SK^{(m+1)}\right)^{3}\right)'W\left(T+SK^{(m+1)}\right)^{3}+\left(\left(T+SK^{(m+1)}\right)^{3}\right)'W\left(T+SK^{(m+1)}\right)^{3}\\
&\qquad-\cdots\\
&\quad=W\left(K^{(m)}\mid T\right).
\end{aligned}$$

Since

$$W\left(K^{(m+1)}\mid T\right)=\sum_{j=0}^{\infty}\left(\left(T+SK^{(m+1)}\right)^{j}\right)'\left(A+\left(K^{(m+1)}\right)'BK^{(m+1)}\right)\left(T+SK^{(m+1)}\right)^{j},$$

this completes the proof. □

Next, we show that the above iterative procedure leads to a solution of (4.16). Since $\{W\left(K^{(i)}\mid T\right), i=0,1,\dots\}$ is a non-increasing sequence of non-negative definite matrices, the limit

$$\lim_{i\to\infty}W\left(K^{(i)}\mid T\right)=W(T)$$

exists, and (4.19) implies the existence of the limit

$$\lim_{i\to\infty}K^{(i)}=K(T).$$

Furthermore, from (4.17) and (4.19), we obtain the formulas

$$W(T) = A + (K(T))' BK(T) + (T + SK(T))' W(T) (T + SK(T)) \tag{4.22}$$

and

$$K(T) = -(B + S' W(T)S)^{-1} S' W(T)T. \tag{4.23}$$

The stability of $\{X_j, j = 0, 1, \dots\}$ under $K(T)$ follows from (4.23) and the proof of Lemma 4.4.2. Also, the validity of (4.16) is ensured by (4.22) and (4.23). Note that $W(T)$ is the unique symmetric matrix satisfying (4.16), since $\tilde{W} \geq W(T)$ and $\tilde{W} \leq W(T)$ both must hold whenever $\tilde{W}$ satisfies (4.16). Consequently, $W(T)$ and $K(T)$ are uniquely determined by (4.22) and (4.23).

For convenience, set $W^0 = W(T^0)$, $K^0 = K(T^0)$, and $\gamma = \mathrm{tr}\,(W^0 Q)$.

Lemma 4.4.3. *Under any stable control policy* $\{K_j, j = 0, 1, \dots\}$,

$$M_k = C_k^{\mathrm{LQ}} - k\gamma + X_k' W^0 X_k - X_0' W^0 X_0 - \sum_{j=0}^{k-1} X_j' \left(K_j - K^0\right)' \left(B + S' W^0 S\right) \left(K_j - K^0\right) \tag{4.24}$$

is a martingale with respect to $\{\mathcal{F}_k, k = 1, 2, \dots\}$. *Moreover,*

$$\lim_{k \to \infty} k^{-1} M_k = 0, \quad \text{a.s.} \tag{4.25}$$

Proof. By (4.22) and (4.23), we have

$$\begin{aligned}
\left(K_j - K^0\right)' \left(B + S'W^0S\right)\left(K_j - K^0\right) + W^0 &= K_j' \left(B + S'W^0S\right) K_j + \left(K^0\right)' \left(B + S'W^0S\right) K^0 \\
&\quad - K_j' \left(B + S'W^0S\right) K^0 - \left(K^0\right)' \left(B + S'W^0S\right) K_j \\
&\quad + A + \left(K^0\right)' BK^0 + \left(T^0 + SK^0\right)' W^0 \left(T^0 + SK^0\right) \\
&= K_j' BK_j + K_j' S'W^0SK_j \\
&\quad \left(K^0\right)' \left(B + S'W^0S\right)\left[-\left(B + S'W^0S\right)^{-1} S'W^0T^0\right] \\
&\quad - K_j' \left(B + S'W^0S\right)\left[-\left(B + S'W^0S\right)^{-1} S'W^0T^0\right] \\
&\quad - \left[-\left(B + S'W^0S\right)^{-1} S'W^0T^0\right]' \left(B + S'W^0S\right) K_j \\
&\quad + A + \left(K^0\right)' BK^0 + \left(T^0 + SK^0\right)' W^0 \left(T^0 + SK^0\right) \\
&= K_j' BK_j + K_j' S'W^0SK_j \\
&\quad - \left(K^0\right)' S'W^0T^0 + K_j' S'W^0T^0 + \left(T^0\right)' \left(W^0\right)' SK_j \\
&\quad + A + \left(K^0\right)' BK^0 + \left(T^0 + SK^0\right)' W^0 \left(T^0 + SK^0\right)
\end{aligned}$$

Observe that

$$\begin{aligned}
&K_j' S'W^0SK_j + K_j' S'W^0T^0 + \left(T^0\right)' \left(W^0\right)' SK_j + \left(T^0 + SK^0\right)' W^0 \left(T^0 + SK^0\right) \\
&\qquad = \left(T^0 + SK_j\right)' W^0 \left(T^0 + SK_j\right) + \left(K^0\right)' S'W^0SK^0 + \left(K^0\right)' S'W^0T^0 + \left(T^0\right)' \left(W^0\right)' SK^0
\end{aligned}$$

and

$$\begin{aligned}
&\left(K^0\right)' BK^0 + \left(K^0\right)' S'W^0 SK^0 + \left(T^0\right)' \left(W^0\right)' SK^0 \\
&\qquad = \left(\left(K^0\right)' \left(B + S'W^0 S\right) + \left(T^0\right)' \left(W^0\right)' S\right) K^0 \\
&\qquad = \left(\left[-\left(B + S'W^0 S\right)^{-1} S'W^0 T^0\right]' \left(B + S'W^0 S\right) + \left(T^0\right)' \left(W^0\right)' S\right) K^0 \\
&\qquad = \left(-\left(T^0\right)' \left(W^0\right)' S + \left(T^0\right)' \left(W^0\right)' S\right) K^0 \\
&\qquad = 0.
\end{aligned}$$

Thus,

$$\left(K_j - K^0\right)' \left(B + S'W^0 S\right) \left(K_j - K^0\right) + W^0 = A + K_j' BK_j + \left(T^0 + SK_j\right)' W^0 \left(T^0 + SK_j\right). \quad (4.26)$$

By (4.24), (4.26), and the definition of C_k^{LQ}, it follows that

$$M_k = \sum_{j=0}^{k-1} Y_j,$$

where

$$\begin{aligned}
Y_j &= X_j' AX_j + U_j' BU_j - \gamma + X_{j+1}' W^0 X_{j+1} - X_j' W^0 X_j - X_j' \left(K_j - K^0\right)' \left(B + S'W^0 S\right) \left(K_j - K^0\right) X_j \\
&= X_j' AX_j + U_j' BU_j - \gamma + X_{j+1}' W^0 X_{j+1} - X_j' \left[A + K_j' BK_j + \left(T^0 + SK_j\right)' W^0 \left(T^0 + SK_j\right)\right] X_j \\
&= X_{j+1}' W^0 X_{j+1} - \left(T^0 X_j + SU_j\right)' W^0 \left(T^0 X_j + SU_j\right) - \gamma \\
&= \left[X_{j+1}' - \left(T^0 X_j + SU_j\right)'\right] W^0 \left[X_{j+1} + \left(T^0 X_j + SU_j\right)\right] - \gamma \\
&= \left[X_{j+1}' - \left(T^0 X_j + SU_j\right)'\right] W^0 \left[\left(X_{j+1} - \left(T^0 X_j + SU_j\right)\right) + 2\left(T^0 X_j + SU_j\right)\right] - \gamma \\
&= \xi_{j+1}' W^0 \left[\xi_{j+1} + 2\left(T^0 X_j + SU_j\right)\right] - \gamma \\
&= \xi_{j+1}' W^0 \xi_{j+1} - \gamma + 2\xi_{j+1}' W^0 \left(T^0 X_j + SU_j\right).
\end{aligned}$$

Hence,

$$\begin{aligned}\mathbb{E}\left[Y_j \mid \mathcal{F}_j\right] &= \mathbb{E}\left[\xi'_{j+1}W^0\xi_{j+1} - \gamma + 2\xi'_{j+1}W^0\left(T^0X_j + SU_j\right) \mid F_j\right] \\ &= \mathbb{E}\left[\xi'_{j+1}W^0\xi_{j+1}\right] - \gamma + 2\mathbb{E}\left[\xi'_{j+1} \mid F_j\right]W^0\left(T^0X_j + SU_j\right) \\ &= \operatorname{tr}\left(W^0Q\right) - \gamma \\ &= 0,\end{aligned}$$

and since Y_j is $\mathcal{F}_{j+1}$-measurable for $j = 0, 1, \dots$, we conclude that $\{M_k, k = 0, 1, \dots\}$ is a martingale with respect to $\{F_k, k = 1, 2, \dots\}$.

Next, we estimate the second moments

$$\begin{aligned}\mathbb{E}\left[Y_j^2 \mid \mathcal{F}_j\right] &= \mathbb{E}\left[\left(\xi'_{j+1}W^0\xi_{j+1} - \gamma + 2\xi'_{j+1}W^0\left(T^0X_j + SU_j\right)\right)^2 \mid \mathcal{F}_j\right] \\ &= \mathbb{E}\left[\left(\xi'_{j+1}W^0\xi_{j+1} - \gamma\right)^2\right] \\ &\quad + 4\left(T^0X_j + SU_j\right)'\mathbb{E}\left[W^0\xi_{j+1}\left(\xi'_{j+1}W^0\xi_{j+1} - \gamma\right)\right] \\ &\quad + 4\left(T^0X_j + SU_j\right)'W^0QW^0\left(T^0X_j + SU_j\right) \\ &= C\left(1 + \mathbb{E}\|T^0X_j + SU_j\|^2\right).\end{aligned}$$

Consequently,

$$\mathbb{E}\left[Y_j^2\right] = \mathbb{E}\left[\mathbb{E}\left[Y_j^2 \mid F_j\right]\right] = C\left(1 + \mathbb{E}\|X_{j+1}\|^2\right), \ j = 0, 1, \dots$$

The stability of $\{X_j, j = 0, 1, \dots\}$ and Corollary 4.1.1 imply the hypothesis of Lemma B.0.1 (Martingale SLLN), from which we obtain (4.25). □

Corollary 4.4.4. *Under any stable control policy $\{K_j, j = 0, 1, \dots\}$, we have*

$$\limsup_{k\to\infty} k^{-1}C_k^{\mathrm{LQ}} \geq \gamma, \quad \text{a.s.}$$

Proof. By (4.24), we have

$$k^{-1}C_k^{\mathrm{LQ}} \geq \gamma + k^{-1}M_k - k^{-1}X_k'W^0X_k,$$

and it follows from (4.25) that

$$\limsup_{k\to\infty} k^{-1}C_k^{\mathrm{LQ}} \geq \gamma - \liminf_{k\to\infty} k^{-1}X_k'W^0X_k.$$

By Fatou's Lemma and (3.2),

$$\begin{aligned}\mathbb{E}\left[\liminf_{k\to\infty} k^{-1}X_k'W^0X_k\right] &\leq \liminf_{k\to\infty} k^{-1}\mathbb{E}\left[X_k'W^0X_k\right] \\ &\leq \|W^0\| \liminf_{k\to\infty} k^{-1}\mathbb{E}\|X_k\|^2 \\ &= 0.\end{aligned}$$

Hence,

$$\liminf_{k\to\infty} k^{-1}X_k'W^0X_k = 0, \quad \text{a.s.},$$

which completes the proof. □

Theorem 4.4.5. *Let* $\{K_j, j = 0, 1, \dots\}$ *be a stable control policy such that*

$$\lim_{j\to\infty} K_j = K^0, \quad \text{a.s.} \tag{4.27}$$

Then, under that policy, we have

$$\lim_{k\to\infty} k^{-1}C_k^{\mathrm{LQ}} = \gamma, \quad \text{a.s.} \tag{4.28}$$

Proof. Lemma 4.4.3 reduces the proof to verifying that

$$\lim_{k\to\infty} k^{-1}X_k'W^0X_k = 0, \quad \text{a.s.} \tag{4.29}$$

and

$$\lim_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} X_j' \left(K_j - K^0\right)' \left(B + S'W^0S\right) \left(K_j - K^0\right) X_j = 0, \quad \text{a.s.} \tag{4.30}$$

By Lemma 4.2.1, we have

$$\sum_{k=1}^{\infty} k^{-2} ||X_k||^4 < \infty, \quad \text{a.s.}$$

Hence

$$\lim_{k\to\infty} k^{-1} ||X_k||^2 = 0, \quad \text{a.s.},$$

which implies (4.29). On the other hand,

$$\begin{aligned}
&\lim_{k\to\infty} k^{-1} \sum_{j=0}^{k-1} X_j' \left(K_j - K^0\right)' \left(B + S'W^0S\right) \left(K_j - K^0\right) X_j \\
&\qquad \le \limsup_{k\to\infty} k^{-1} ||B + S'W^0S|| \sum_{j=0}^{k-1} ||K_j - K^0||^2 \, ||X_j||^2 \\
&\qquad \le \limsup_{k\to\infty} ||B + S'W^0S|| \left(k^{-1} \sum_{j=0}^{k-1} \left|\left|K_j - K^0\right|\right|^4\right)^{1/2} \left(k^{-1} \sum_{j=0}^{k-1} ||X_j||^4\right)^{1/2},
\end{aligned}$$

which equals 0 a.s., in view of (4.27) and Lemma 4.2.1. □

The following corollary is a direct consequence of Corollary 4.4.4 and Theorem 4.4.5.

Corollary 4.4.6. *The stationary control K^0 is optimal in the sense of* (4.2) *and* (4.3).

The following lemma asserts that $K(T)$ is a smooth function of T. The proof, which is omitted here, relies on the implicit function theorem; the details may be found in [43].

Lemma 4.4.7. *Let T^0 be such that Assumptions 4.1 and 4.2 are satisfied. Then, in a neighborhood of T^0, the matrix $K(T)$ is differentiable arbitrarily-many times with respect to T.*

We now return to the case where the teleportation matrix, T, depends on an unknown parameter, α. We shall write $K^{\mathrm{LQ}}(\alpha) = K(T(\alpha))$. Consider a control policy that is constructed as follows. At time 0, we select $\alpha_0^* \in \mathcal{A}_\alpha$ and set $K_0 = K^{\mathrm{LQ}}(\alpha_0^*)$. For $j = 1, 2, \ldots$, we

observe $\{X_0, \ldots, X_j\}$, let the estimate $\hat{\alpha}_j$ be the minimizer of the least squares functional (3.3) (or (3.16)), and set

$$K_j^{\mathrm{LQ}} = K^{\mathrm{LQ}}\left(\alpha_j^*\right),$$

where α_j^* is defined by (3.15). The matrices K_j^{LQ}, $j = 0, 1, \ldots$, are estimates of the optimal stationary control, $K^0 = K^{\mathrm{LQ}}(\alpha_0)$.

As a consequence of Lemma 4.4.7, we may use the Taylor expansion to write

$$K^{\mathrm{LQ}}(\alpha) = K^{\mathrm{LQ}}(\alpha_0) + \sum_{i=1}^{q} \left(\alpha^{(i)} - \alpha_0^{(i)}\right) \frac{\partial}{\partial \alpha^{(i)}} K^{\mathrm{LQ}}(\alpha_0) + O\left(\|\alpha - \alpha_0\|^2\right).$$

Thus, it follows from Corollary 3.2.3 (or Corollary 3.2.3, respectively) that

$$K_j^{\mathrm{LQ}} = K^{\mathrm{LQ}}\left(\alpha_j^*\right) \xrightarrow{\text{a.s.}} K^{\mathrm{LQ}}(\alpha_0) = K^0 \text{ as } j \to \infty. \tag{4.31}$$

The following regularity conditions are assumed in order to ensure that, for each $\alpha \in \mathcal{A}_\alpha$, the optimal stationary control $K^{\mathrm{LQ}}(\alpha) = K(T(\alpha))$ can be calculated via Howard's dynamic programming techniques as shown above.

Assumption 4.3. *For all $\alpha \in \mathcal{A}_\alpha$, Assumption 4.1 is satisfied with $T = T(\alpha)$.*

Assumption 4.4. *For all $\alpha \in \mathcal{A}_\alpha$, Assumption 4.2 is satisfied with $T = T(\alpha)$.*

In [43], Mandl observes that Assumptions 4.3 and 4.4 imply that Assumption 3.1 is satisfied locally; that is, when K is taken from a neighborhood of $K(\alpha)$, a V for which (3.4) holds can always be found.

Our final theorem asserts that the control policy constructed above is equivalent to the optimal stationary control with regards to the long term average of the quadratic cost criterion. This result is an immediate consequence of (4.31) and Theorem 4.4.5.

Theorem 4.4.8. *Under the control policy $\left\{K_j^{\mathrm{LQ}}, j = 0, 1, \ldots\right\}$, we have*

$$\lim_{k \to \infty} k^{-1} C_k^{\mathrm{LQ}} = \gamma, \quad \text{a.s.} \tag{4.32}$$

4.5 Simulations

We now present simulations of the various control policies described in the preceding sections of this chapter. As before, for each example, the web graph is randomly generated according to the procedure described in Section 2.3. Moreover, we assume that $R = I$ and that the noise terms, $\{\xi_j, j = 0, 1, \dots\}$, are i.i.d. with zero mean and covariance matrix $Q = \sigma^2 I$, where $\sigma^2 = 10^{-5}$.

Example 4.1 (Minimum Variance Control)**.** Consider a web of $n = 10$ pages and 27 hyperlinks. We assume that the corresponding stochastic system model for PageRank depends on a one-dimensional unknown parameter α. We employ the minimum variance control of Section 4.2.

In Figure 4.1, we see the sequence of PageRank iterates under the minimum variance control policy. As expected, the effect of this type of control is to drive all PageRank values to zero, with the slight deviations from zero after the first few iterations being caused by the random noise.

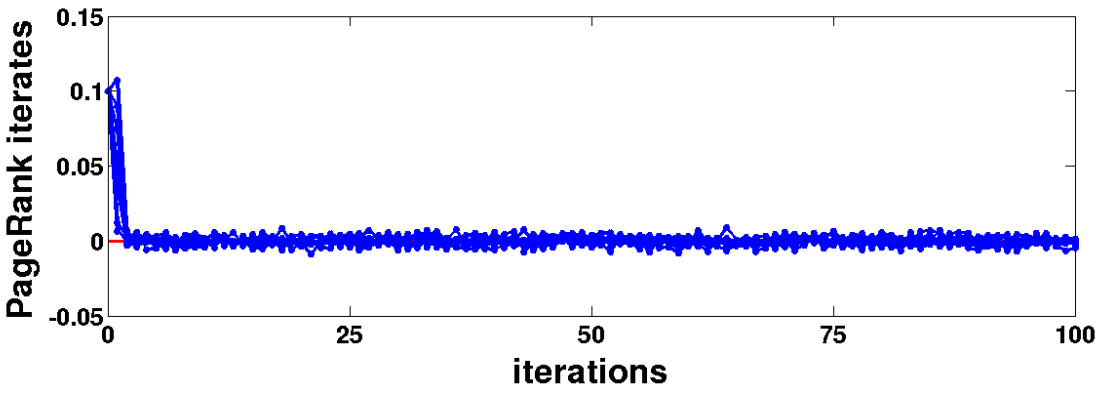


Figure 4.1: Controlled PageRank iterates for Ex. 4.1

In Figure 4.2, we observe the convergence of the least squares parameter estimates to the true value $\alpha_0 = 1/2$ under this control scheme.

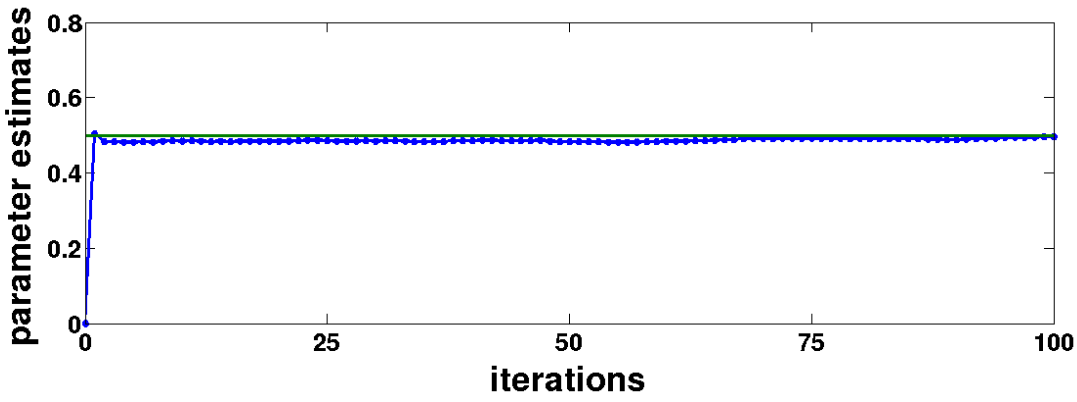


Figure 4.2: Least squares parameter estimation for Ex. 4.1

Figure 4.3 illustrates the limiting behavior of the time-averaged minimum variance cost criterion. In particular, we see that $\lim_{k\to\infty} k^{-1}C_k^{\mathrm{MV}} = \gamma$, where

$$\begin{aligned}\gamma &= \mathrm{tr}(Q)\\ &= n\sigma^2\\ &= 10^{-4}.\end{aligned}$$

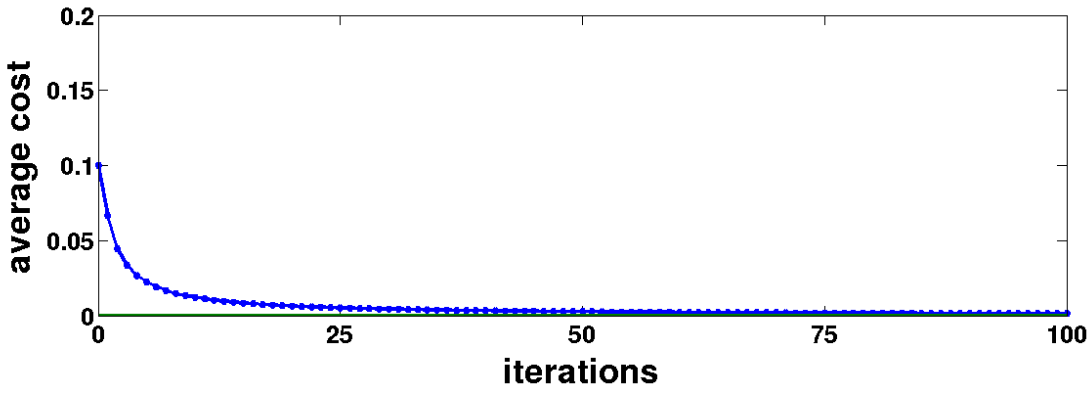


Figure 4.3: Limiting behavior of average cost for Ex. 4.1

Finally, in Figure 4.4, we see that the normed difference between the estimated optimal control, K_j^{MV}, and the optimal stationary control, K^0, goes to zero almost surely as $j \to \infty$.

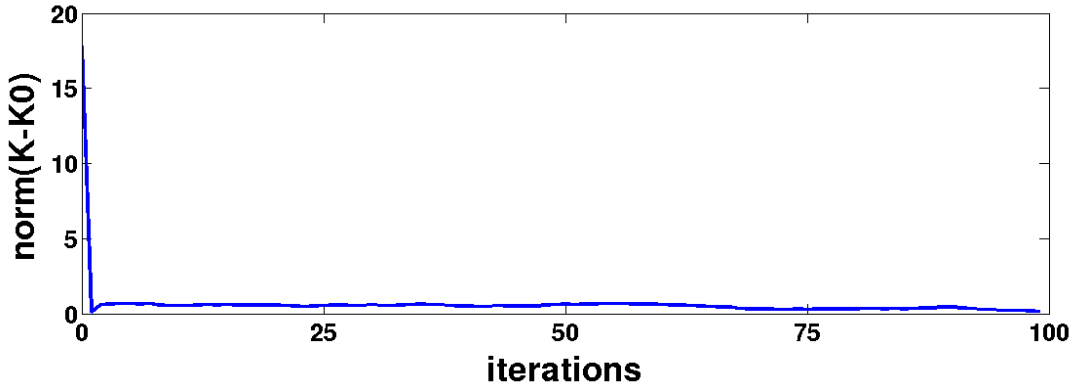


Figure 4.4: Limiting behavior of $\left\{\left\|K_j^{\mathrm{MV}} - K^0\right\|, j = 0, 1, \dots\right\}$ for Ex. 4.1

Example 4.2 (Adaptive Tracking)**.** Consider a web of $n = 10$ pages and 21 hyperlinks. We assume that the corresponding stochastic system model for PageRank depends on a one-dimensional unknown parameter α. We employ the adaptive tracking strategy of Section 4.3, where our reference signal is the constant vector

$$Y = (0.0311, 0.0763, 0.2063, 0.1148, 0.1414, 0.1340, 0.1605, 0.0067, 0.1213, 0.0074).$$

In Figure 4.5, we see the sequence of PageRank iterates under the adaptive tracking control policy. As remarked previously, and as suggested by the name, the effect of this type of control is to cause the PageRank values to track the distribution of values given by Y.

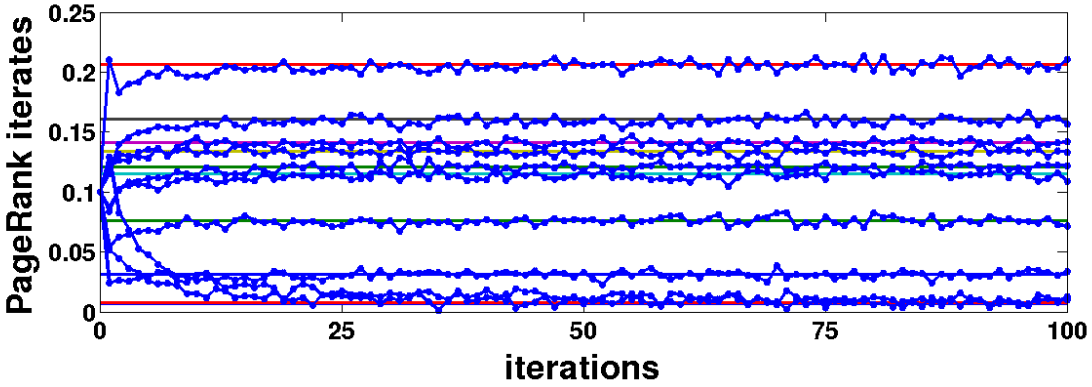


Figure 4.5: Controlled PageRank iterates for Ex. 4.2

In Figure 4.6, we observe the convergence of the least squares parameter estimates to the true value $\alpha_0 = 1/2$ under this control scheme.

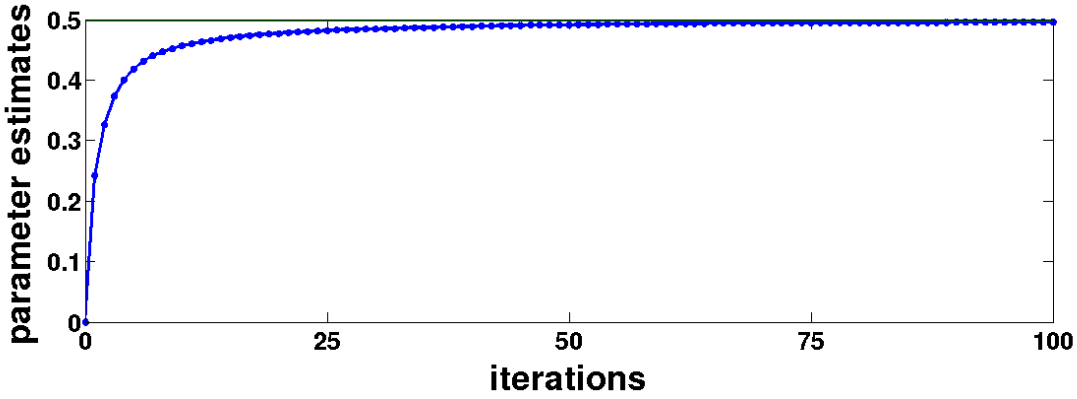


Figure 4.6: Least squares parameter estimation for Ex. 4.2

Figure 4.7 illustrates the limiting behavior of the time-averaged minimum variance cost criterion. In particular, we see that $\lim_{k\to\infty} k^{-1} C_k^{\mathrm{AT}} = \gamma$, where again $\gamma = 10^{-4}$.

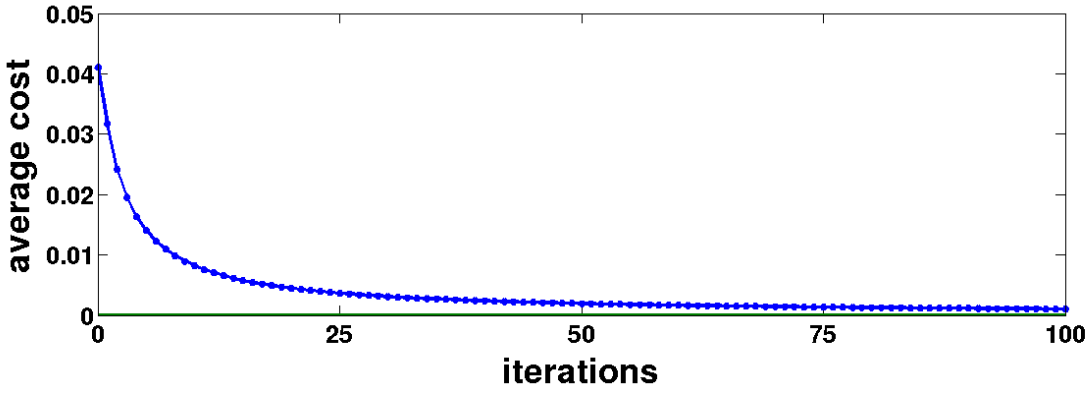


Figure 4.7: Limiting behavior of average cost for Ex. 4.2

Finally, in Figure 4.8, we see that the normed difference between the estimated optimal control, K_j^{AT}, and the optimal stationary control, K^0, goes to zero almost surely as $j \to \infty$.

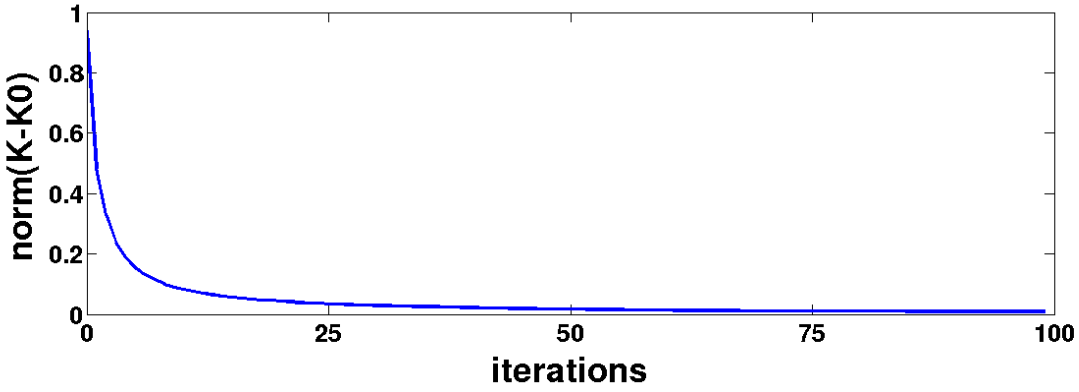


Figure 4.8: Limiting behavior of $\left\{\left\|K_j^{\mathrm{AT}} - K^0\right\|, j = 0, 1, \dots\right\}$ for Ex. 4.2

Example 4.3 (Quadratic Cost Criterion)**.** Consider a web of $n = 10$ pages and 25 hyperlinks. We assume that the corresponding stochastic system model for PageRank depends on a one-dimensional unknown parameter α. We assume a quadratic cost criterion with $A = B = I$ and construct a control policy via the dynamic programming approach of Section 4.4.

In Figure 4.9, we see the sequence of PageRank iterates under the quadratic cost control policy. As in the case of the minimum variance policy, the effect of this type of control is to drive all PageRank values to zero.

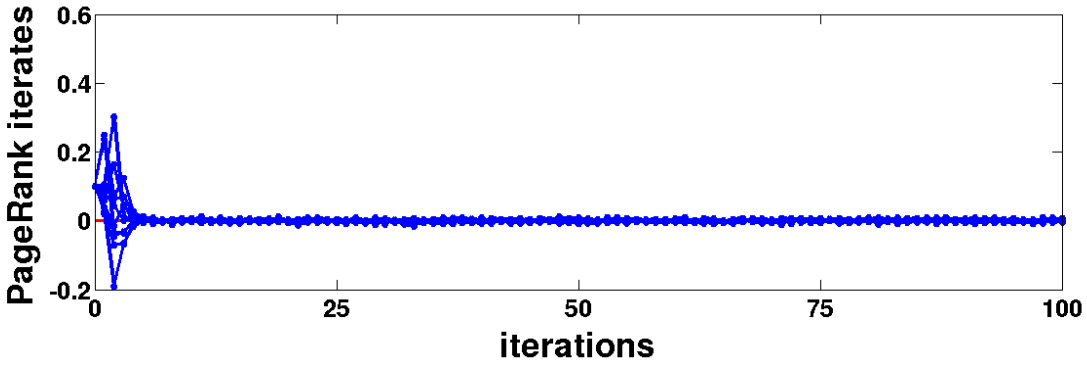


Figure 4.9: Controlled PageRank iterates for Ex. 4.3

In Figure 4.10, we observe the convergence of the least squares parameter estimates to the true value $\alpha_0 = 1/2$ under this control scheme.

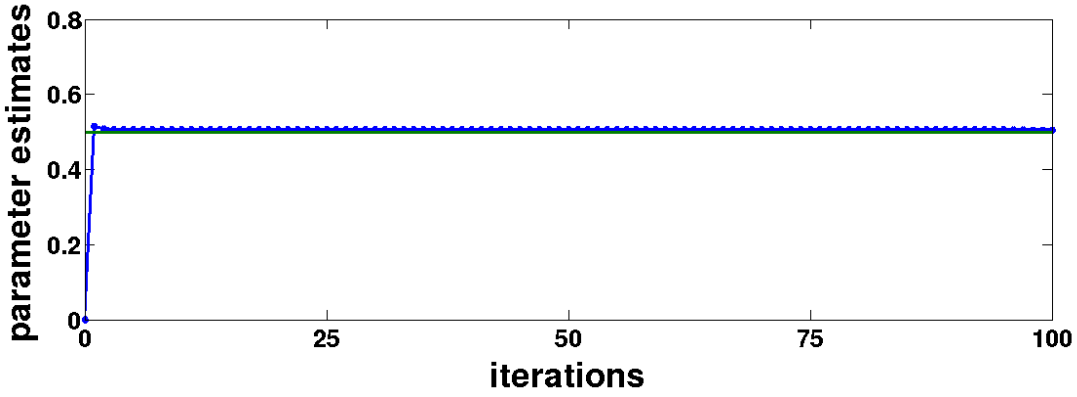


Figure 4.10: Least squares parameter estimation for Ex. 4.3

Figure 4.11 illustrates the limiting behavior of the time-averaged minimum variance cost criterion. In particular, we see that $\lim_{k\to\infty} k^{-1}C_k^{\mathrm{LQ}} = \gamma$, where

$$\begin{aligned}\gamma &= \mathrm{tr}(WQ)\\ &= n\sigma^2\mathrm{tr}(W)\\ &\approx 4.6327\times 10^{-4}.\end{aligned}$$

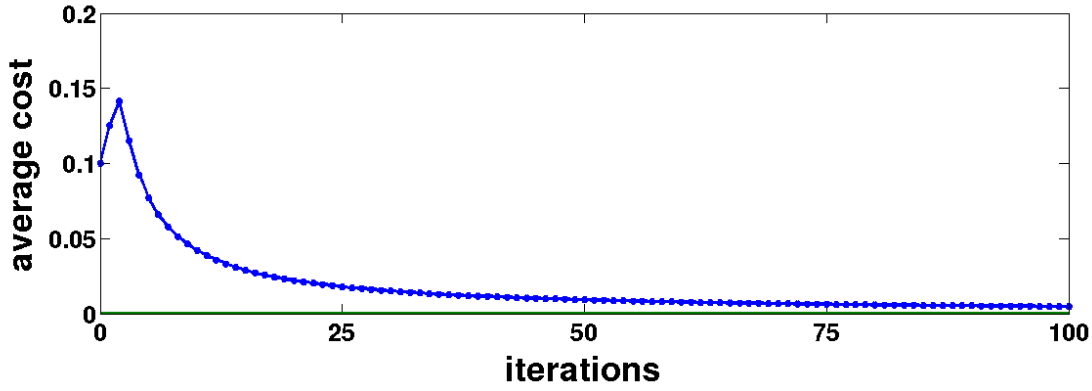


Figure 4.11: Limiting behavior of average cost for Ex. 4.3

Finally, in Figure 4.12, we see that the normed difference between the estimated optimal control, K_j^{LQ}, and the optimal stationary control, K^0, goes to zero almost surely as $j \to \infty$.

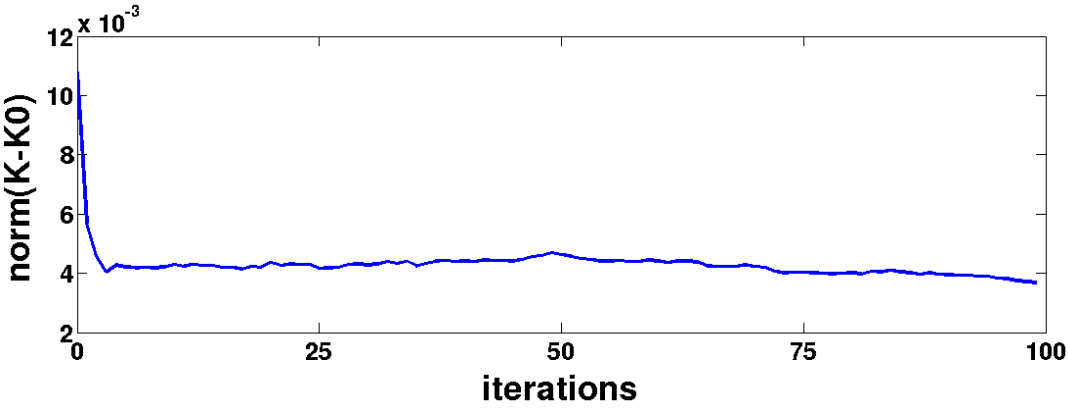


Figure 4.12: Limiting behavior of $\left\{\left\|K_j^{\mathrm{LQ}} - K^0\right\|, j = 0, 1, \dots\right\}$ for Ex. 4.3

Chapter 5

Final Remarks

5.1 Contributions

This dissertation has documented a stochastic system reformulation of the PageRank problem that lies at the heart of the Google web search engine. In particular:

- We presented a model for PageRank whose dynamics are described by a controlled stochastic system that depends on an unknown parameter.

- We established strong consistency of a least squares estimator for the unknown parameter in the stochastic system. Furthermore, motivated by recent work on distributed randomized methods for PageRank computation, we showed that the least squares estimator remains strongly consistent within a distributed framework.

- We considered the problem of adaptively controlling the stochastic system model for PageRank. Under three different cost criteria, we applied a control procedure that proceeds simultaneously with the least squares estimation of the unknown parameter. Specifically, we used the parameter estimates to iteratively construct an adaptive control policy whose performance, according to the long-run average cost, is equivalent to the optimal stationary control that would be used if we had knowledge of the true value of the parameter.

This research has been accepted for presentation at two international conferences:

- 27th International Federation for Information Processing (IFIP) TC7 Conference on System Modelling and Optimization, Sophia Antipolis, France, June 29 - July 3, 2015.
- Society for Industrial and Applied Mathematics (SIAM) Conference on Control and its Applications (CT15), Paris, France, July 8 - 10, 2015.

An accompanying paper will appear in the proceedings of the latter conference:

- C. E. Clifton and B. Pasik-Duncan. *Parameter Estimation in a Stochastic System Model for PageRank*. Proc. SIAM Conf. Control & Appl., Paris, France, to appear.

Moreover, an additional paper is in preparation:

- C. E. Clifton and B. Pasik-Duncan. *Adaptive Control of a Stochastic System Model for PageRank*. To be submitted.

5.2 Future Work

This research has laid the groundwork to explore many more interesting problems in the area of stochastic adaptive control applied to the PageRank algorithm. Among them:

- To develop more sophisticated numerical simulations, in order to practically apply the results presented herein on the scale of the massive and ever-growing World Wide Web.
- To consider more general procedures, such as weighted least squares, for estimating the true value of the unknown parameter upon which the stochastic system model for PageRank is assumed to depend.
- To consider other types of control policies, such as those generated by stochastic differential equations.

5.3 Conclusion

We have provided original insights about the PageRank problem from the viewpoint of stochastic systems theory by demonstrating how stochastic adaptive control principles can be applied to this application of modern interest and relevance.

References

[1] How Search Works - The Story - Inside Search - Google. `https://www.google.com/intl/en_us/insidesearch/howsearchworks/thestory/index.html`, Apr 2015.

[2] A. Arasu, J. Novak, A. Tomkins, and J. Tomlin. PageRank computation and the structure of the web: Experiments and algorithms. Poster at 11th International WWW Conference, 2002.

[3] K. J. Åström, G. C. Goodwin, and P. R. Kumar. *Adaptive Control, Filtering, and Signal Processesing*, volume 74 of *The IMA Volumes in Mathematics and its Applications*. Springer, New York, 1995.

[4] K. J. Åström and R. M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton Univ. Press, 2008.

[5] K. J. Åström and B. Wittenmark. *Adaptive Control*. Dover, 2nd edition, 2008.

[6] K. Avrachenkov, N. Litvak, D. Nemirovsky, and N. Osipova. Monte Carlo methods in PageRank computation: When one iteration is sufficient. *SIAM J. Numer. Anal.*, 45(2):890–904, Feb 2007.

[7] R. Baeza-Yates and E. Davis. Web page ranking using link attributes. In *Proc. 13th International WWW Conference on Alternate Track Papers & Posters*, WWW Alt. '04, pages 328–329, New York, NY, USA, 2004. ACM.

[8] R. F. Bass. *Stochastic Processes*. Cambridge Univ. Press, New York, NY, USA, 2011.

[9] S. Brin, R. Motwani, L. Page, and T. Winograd. What can you do with a web in your pocket? *IEEE Data Eng. Bull.*, 21(2):37–47, Jun 1998.

[10] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, Apr 1998.

[11] A. Z. Broder, R. Lempel, F. Maghoul, and J. Pedersen. Efficient PageRank approximation via graph aggregation. *Inf. Retr.*, 9(2):123–138, Mar 2006.

[12] K. Bryan and T. Leise. The $25,000,000,000 eigenvector: The linear algebra behind Google. *SIAM Rev.*, 48(3):569–581, Mar 2006.

[13] H-F. Chen and L. Guo. *Identification and Stochastic Adaptive Control*. Birkhäuser Boston, 1991.

[14] M. H. A. Davis and R. B. Vinter. *Stochastic Modelling and Control*. Chapman & Hall, New York, NY, USA, 1985.

[15] T. E. Duncan and B. Pasik-Duncan. Stochastic Adaptive Control. In *Encyclopedia of Systems and Control*, pages 1–6. Springer London, Apr 2014.

[16] M. Franceschet. PageRank: Standing on the shoulders of giants. *Commun. ACM*, 54(6):92–101, Jun 2011.

[17] G. Grimmett and D. Stirzaker. *Probability and Random Processes*. Oxford University Press, New York, NY, USA, 3rd edition, 2001.

[18] T. H. Haveliwala. Topic-sensitive PageRank. In *Proc. 11th International WWW Conference*, WWW '02, pages 517–526, New York, NY, USA, 2002. ACM.

[19] T. H. Haveliwala and S. D. Kamvar. The second eigenvalue of the Google matrix. Technical Report 2003-20, Stanford InfoLab, Mar 2003.

[20] T. H. Haveliwala, S. D. Kamvar, and G. Jeh. An analytical comparison of approaches to personalizing PageRank. Technical Report 2003-35, Stanford InfoLab, Jun 2003.

[21] R. A. Howard. *Dynamic Programming and Markov Processes*. MIT Press / Wiley, New York, NY, USA, 1960.

[22] I. C. F. Ipsen and T. M. Selee. PageRank computation, with special attention to dangling nodes. *SIAM J. Matrix Anal. & Appl.*, 29(4):1281–1296, Dec 2007.

[23] H. Ishii and R. Tempo. Computing the PageRank variation for fragile web data. *SICE J. of Control, Measurement, and System Integration*, 2(1):1–9, Jan 2009.

[24] H. Ishii and R. Tempo. Distributed randomized algorithms for PageRank computation. *IEEE Trans. Autom. Control*, 55(9):1987–2002, Sep 2010.

[25] H. Ishii and R. Tempo. The PageRank problem, multi-agent consensus and web aggregation: A systems and control viewpoint. *IEEE Control Sys. Mag.*, 34(3):34–53, Jun 2014.

[26] H. Ishii, R. Tempo, and E-W. Bai. A web aggregation approach for distributed randomized PageRank algorithms. *IEEE Trans. Autom. Control*, 57(11):2703–2717, Nov 2012.

[27] H. Ishii, R. Tempo, and E-W. Bai. PageRank computation via a distributed randomized approach with lossy communication. *Systems & Control Letters*, 61(12):1221–1228, Dec 2012.

[28] G. Jeh and J. Widom. Scaling personalized web search. In *Proc. 12th International WWW Conference*, pages 271–279, 2003.

[29] S. D. Kamvar and T. H. Haveliwala. The condition number of the PageRank problem. Technical Report 2003-36, Stanford InfoLab, Jun 2003.

[30] S. D. Kamvar, T. H. Haveliwala, and G. H. Golub. Adaptive methods for the computation of PageRank. *Linear Algebra & Appl.*, 386:51–65, Jul 2004.

[31] S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub. Exploiting the block structure of the web for computing PageRank. Technical Report 2003-17, Stanford InfoLab, Mar 2003.

[32] S. D. Kamvar, T. H. Haveliwala, C. D. Manning, and G. H. Golub. Extrapolation methods for accelerating PageRank computations. In *Proc. 12th International WWW Conference*, WWW '03, pages 261–270, New York, NY, USA, 2003. ACM.

[33] P. R. Kumar and P. Varaiya. *Stochastic Systems: Estimation, Identification, and Adaptive Control*. Prentice-Hall, Englewood Cliffs, NJ, USA, 1986.

[34] A. N. Langville and C. D. Meyer. Updating PageRank using the group inverse and stochastic complementation. Technical Report crsc02-tr32, NC State CRSC, 2002.

[35] A. N. Langville and C. D. Meyer. Deeper inside PageRank. *Internet Math.*, 1(3):335–380, 2004.

[36] A. N. Langville and C. D. Meyer. Updating Pagerank with iterative aggregation. In *Proc. 13th International WWW Conference on Alternate Track Papers & Posters*, WWW Alt. '04, pages 392–393, New York, NY, USA, 2004. ACM.

[37] A. N. Langville and C. D. Meyer. A reordering for the PageRank problem. *SIAM J. Sci. Comput.*, 27(6):2112–2120, Dec 2006.

[38] A. N. Langville and C. D. Meyer. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton Univ. Press, Princeton, NJ, USA, 2006.

[39] A. N. Langville and C. D. Meyer. Updating Markov chains with an eye on Google's PageRank. *SIAM J. Matrix Anal. & Appl.*, 27(4):968–987, 2006.

[40] G. F. Lawler. *Introduction to Stochastic Processes*. Chapman & Hall/CRC, Boca Raton, FL, USA, 2nd edition, 2006.

[41] C. P. Lee, G. H. Golub, and S. A. Zenios. A fast two-stage algorithm for computing PageRank and its extensions. Technical report, Stanford InfoLab, 2003.

[42] C. P. Lee, G. H. Golub, and S. A. Zenios. A two-stage algorithm for computing PageRank and multistage generalizations. *Internet Math.*, 4(4):299–327, 2007.

[43] P. Mandl. The use of optimal stationary policies in the adaptive control of linear systems. In *Proc. Symp. to Honour J. Neyman*, pages 223–242, 1974.

[44] C. D. Meyer. *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, Jun 2000.

[45] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, Nov 1999.

[46] G. Pandurangan, P. Raghavan, and E. Upfal. Using PageRank to characterize web structure. *Internet Math.*, 3(1):1–20, 2006.

[47] B. Pasik-Duncan. *On Adaptive Control*. SGPiS-Publishers, Warsaw, 1986.

[48] M. Richardson and P. Domingos. The intelligent surfer: Probabilistic combination of link and content information in PageRank. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

[49] A. Das Sarma, A. R. Molla, G. Pandurangan, and E. Upfal. Fast distributed PageRank computation. In *Distributed Computing and Networking*, volume 7730 of *Lecture Notes in Computer Science*, pages 11–26. Springer Berlin Heidelberg, 2013.

[50] T. Söderström. *Discrete-time Stochastic Systems: Estimation and Control*. Advanced Textbooks in Control and Signal Processing. Springer-Verlag, London, 2nd edition, 2002.

[51] W. Zhao, H-F. Chen, and H-T. Fang. Convergence of distributed randomized PageRank algorithms. *IEEE Trans. Autom. Control*, 58(12):3255–3259, Dec 2013.

[52] Y. Zhu, S. Ye, and X. Li. Distributed PageRank computation based on iterative aggregation-disaggregation methods. In *Proc. 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 578–585, New York, NY, USA, 2005. ACM.

Appendix A

Markov Chains

This appendix is intended to complement Chapter 2 by providing an elementary introduction to the theory of discrete-time Markov chains on a finite state space. For more on Markov chains and stochastic processes in general, we refer to the text of Lawler [40].

The term *Markov chain* classically refers to a discrete-time stochastic process characterized by the Markov property. This property, which will be formalized below, asserts that the next state of the process depends only on the current state and not on any of the previous states. Continuous-time Markov processes also exist, but will not be considered here. Moreover, we will henceforth assume that a Markov chain defines a system with a finite state space, except where otherwise specified.

The word "chain" hints at the discrete nature of the time-dependancy of the process, as well as the particular type of *memorylessness* implied by the Markov property. A Markov chain can be thought of as a random walk between the states of a system. Just as the links of a physical chain are connected to one another in a serial manner, so too do we consider two states to be linked by the random walk if there is a positive probability of moving between them in at least one direction.

The time associated with a Markov chain, as in discrete-*time*, may also be thought of as a physical distance or, in fact, any discretely-measurable quantity. As with any stochastic

process, the theory is unaffected by what "time" actually refers to. The important thing is that, formally, the steps of a Markov chain are indexed by a countable set (commonly the natural numbers) and the chain defines a mapping from that set to the state space.

In view of this description of a mapping from an index set to a state space, we can think of a Markov chain as a sequence of random variables. In particular, at a given time, the state of the system governed by the Markov chain is determined by the value of the random variable associated with that time. This leads to the formal definition.

Definition A.1. A sequence of random variables $\{X_0, X_1, \dots\}$ is a *Markov chain* if and only if

$$\mathbb{P}(X_{n+1} = x_{n+1} \mid X_0 = x_0, X_1 = x_1, \dots, X_n = x_n) = \mathbb{P}(X_{n+1} = x_{n+1} \mid X_n = x_n),$$

where $\mathbb{P}(\cdot \mid \cdot)$ denotes conditional probability and the values $x_0, x_1, \dots, x_{n+1}$ are not necessarily distinct.

The set of possible values of X_i is the *state space* of the chain and is denoted by S. We will let $N \coloneqq |S|$ represent the cardinality of S. There is assumed to be an underlying probability space $(\Omega, \mathcal{F}, \mathbb{P})$ such that each X_n is a $\mathcal{F}$-measurable function that maps Ω into S.

The changes between different states of the system are called *transitions*. Naturally, then, the probabilities with which each of these changes occur are called *transition probabilities*. In the notation of Definition A.1, the probability of transitioning from state i to state j is denoted by $\mathbb{P}(X_{n+1} = j \mid X_n = i)$. This may be abbreviated as p_{ji} when the conditional probability depends only on the current state and next state, i and j, respectively, and not on the current time, n. Such a time-homogeneity property will be assumed throughout.

A Markov chain is completely determined by its state space, its initial state (or initial distribution across the state space), and the collection of its transition probabilities. The latter is commonly represented in the form of a matrix, $P = (p_{ij})_{i,j \in S}$, called the *transition*

matrix. By construction, P is a (column) stochastic matrix, since

$$0 \leq p_{ij} \leq 1, \quad i, j \in S, \tag{A.1}$$

$$\sum_{i \in S} p_{ij} = 1, \quad i \in S. \tag{A.2}$$

Conversely, any matrix whose entries satisfy (A.1) and (A.2) can be the transition matrix for some Markov chain.

A Markov chain does not terminate, in the sense that some "next state" may always be reached after the current state. However, the next state and the current state may eventually always be the same; in this case, we say that the chain has reached an absorbing state. Even if an absorbing state is never reached, it may be possible to classify the long-term behavior of the chain in terms of a so-called stationary distribution. See Section A.1.4 for details.

A.1 Markov chain concepts and properties

Recall that p_{ji} denotes the probability of transitioning directly from state i to state j. This is considered to be a one-step transition probability. More generally, the n-step transition probability defined by

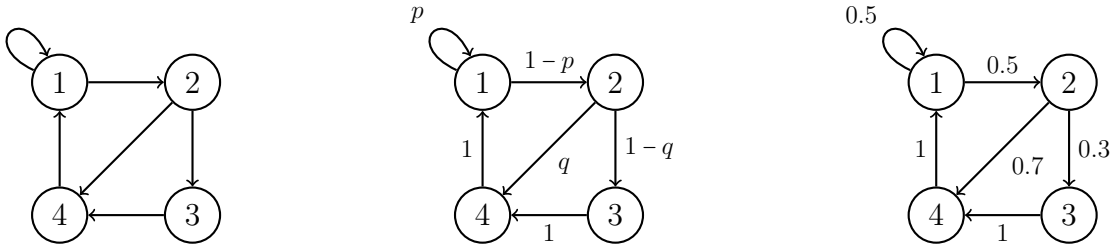
$$p_{ji}^{(n)} := \mathbb{P}(X_n = j \mid X_0 = i)$$

represents to the probability of transitioning from state i to state j over the course of exactly n steps.

The transition probabilities of a Markov chain are conveniently illustrated by way of a *transition diagram*, which is *directed graph*, or *digraph*, whose vertices correspond to the states of the chain and whose directed edges represent possible transitions between states. In particular, the existence of a directed edge from vertex i to vertex j implies that $p_{ji} > 0$.

The edges may be unlabeled or labeled, as shown in Figure A.1. In the latter case, the labeling may either be in terms of arbitrary positive probabilities p and q (Figure A.1b) or

in terms of explicit numerical values (Figure A.1c), provided that, for every i, the sum of all labels corresponding to the edges originating from vertex i is equal to one.



(a) Unlabeled (b) Labeled (c) Labeled

Figure A.1: Unlabeled versus labeled Markov chains

A.1.1 Accessibility and reducibility

A state j is *accessible* from a state i, denoted by $i \to j$, if a system started at state i has a positive probability of reaching state j at some time through some finite sequence of steps. Formally, $i \to j$ if and only if there exists a nonnegative integer n_{ji}, depending on both i and j, such that

$$p_{ji}^{(n_{ji})} = \mathbb{P}(X_{n_{ji}} = j \mid X_0 = i) > 0.$$

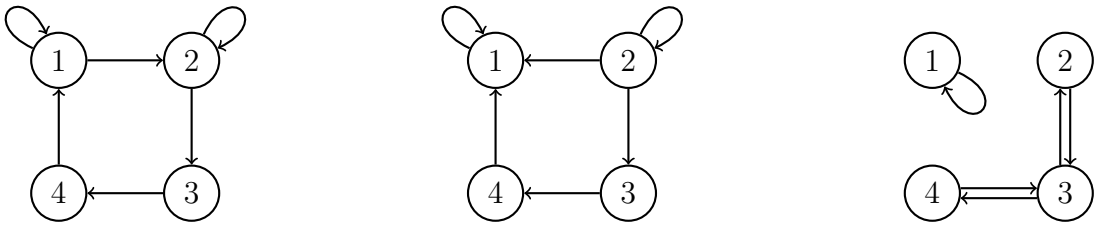
By convention, we assume $n_{ii} = 0$ for all $i \in S$, which implies the intuitive fact that every state is accessible from itself.

A state i *communicates* with a state j, denoted by $i \leftrightarrow j$, if and only if both $i \to j$ and $j \to i$. A set of states C is called a *communicating class* if every pair of states in C communicates and no state in C communicates with a state not in C. Communication of states is an equivalence relation and the communicating classes are the equivalence classes of this relation.

A Markov chain is said to be *irreducible* if its entire state space is a communicating class; in other words, if every state can be reached from every other state through some finite sequence of steps. Otherwise, it is *reducible*.

Figure A.2a illustrates a Markov chain that is irreducible. On the other hand, the chain

in Figure A.2b appears very similar, but is reducible, since it is not possible to reach any other state from state 1. A different sort of reducible chain is shown in Figure A.2c. In graph theoretic terms, the state diagram of an irreducible Markov chain is a strongly connected digraph, while the diagram of a reducible chain is a digraph that is not strongly connected.



(a) Irreducible (b) Reducible (c) Reducible

Figure A.2: Irreducible versus reducible Markov chains

State 1 in both Figures A.2b and A.2c is called an *absorbing state*, since $p_{11} = 1$ and $p_{i1} = 0$ for $i \neq 1$. A Markov chain is said to be a *absorbing Markov chain* if every state can reach an absorbing state; thus, the chain in Figure A.2b is an absorbing Markov chain, whereas the chain Figure A.2c is not.

A.1.2 Periodicity

A state i has *period* $d_i \geq 1$ if a chain starting at i may only return there after some multiple of d_i time steps. In other words, the period d_i is defined by

$$d_i := \gcd\left\{n \geq 1 : p_{ii}^{(n)} > 0\right\},$$

where "gcd" refers to the greatest common divisor. If $d_i = 1$, then the state i is *aperiodic*. Otherwise, it is *periodic with period* d_i.

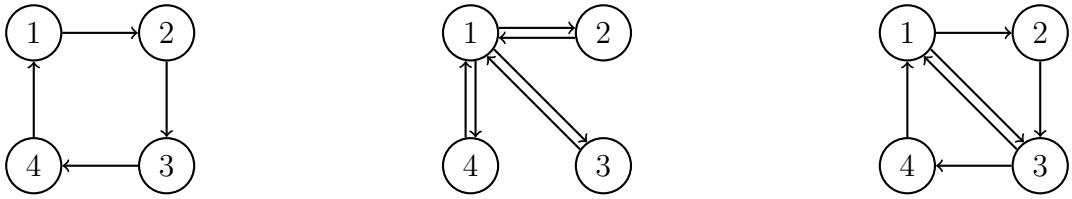
A Markov chain is said to be *aperiodic* if every one of its states is aperiodic. If a Markov chain is irreducible, then the aperiodicity of a single state implies the aperiodicity of the entire chain.

More generally, if two states i and j belong to the same communicating class, then their periods d_i and d_j must coincide. Indeed, if $p_{ii}^{(k)} > 0$, $p_{ji}^{(m)} > 0$, $p_{ij}^{(n)} > 0$ for some integers $k, m, n \geq 1$, then $p_{jj}^{(m+n)} > 0$ and $p_{jj}^{(k+m+n)} > 0$, which implies that

$$d_j \mid (m+n) \text{ and } d_j \mid (k+m+n) \implies d_j \mid [(k+m+n)-(m+n)] \implies d_j \mid k \implies d_j \leq d_i,$$

where $a \mid b$ denotes that a divides b. By symmetry, an analogous argument shows that $d_i \leq d_j$, so we conclude that $d_i = d_j$.

Figures A.3a and A.3b illustrate Markov chains that are periodic, with periods 4 and 2, respectively. On the other hand, the chain in Figure A.3c is aperiodic, since it is irreducible and State 1 is aperiodic.



(a) Periodic (b) Periodic (c) Aperiodic

Figure A.3: Periodic versus aperiodic Markov chains

Any irreducible Markov chain with a positive probability of returning directly from at least one state to itself, illustrated graphically as a self-loop, is aperiodic. Thus, the chains in Figures A.1 and A.2a are aperiodic. If the chain is reducible, aperiodicity is not as easy to check, but it can be verified that the chain in Figure A.2b is aperiodic, while the chain in Figure A.2c is not. Note, however, that it would be incorrect to conclude that the latter chain is periodic, since state 1 has period 1 and states 2 - 4 each have period 2.

A.1.3 Transience and recurrence

A state i is *transient* if a chain starting at i has a positive probability of never returning there. To formalize this notion, we introduce the *hitting time* T_i of the state i, defined by

$$T_i := \inf \{n \geq 1 : X_n = i \mid X_0 = i\}.$$

The random variable T_i represents the first time that a chain started at state i returns to i. Then, i is transient if and only if

$$\mathbb{P}(T_i < \infty) = \sum_{n=1}^{\infty} \mathbb{P}(T_i = n) < 1.$$

A state that is not transient is said to be *recurrent*. In other words, a recurrent state is one with a finite hitting time.

However, if S is allowed to be countably infinite, there are two different types of recurrence that must be distinguished. To this end, we introduce the *mean recurrence time* of a state i, which is the expected value of the hitting time T_i, defined by

$$M_i := \mathbb{E}[T_i] = \sum_{n=1}^{\infty} n \, \mathbb{P}(T_i = n).$$

The state i is *positive recurrent* if $M_i < \infty$, whereas it is *null recurrent* otherwise. Note the distinction between recurrence and positive/null recurrence. For instance, if a state is null recurrent, then it must also be recurrent, meaning that its hitting time is finite with probability one, but at the same time the expectation of that hitting time is infinite. We summarize the various types of recurrence as follows.

State i is *recurrent.*	$\iff$	$\mathbb{P}(T_i < \infty) = 1.$
State i is *positive recurrent.*	$\iff$	$\mathbb{P}(T_i < \infty) = 1$ and $\mathbb{E}[T_i] < \infty.$
State i is *null recurrent.*	$\iff$	$\mathbb{P}(T_i < \infty) = 1$ and $\mathbb{E}[T_i] = \infty.$

An equivalent definition of positive recurrence and null recurrence, which suggests that these concepts relate to the long-term behavior of a Markov chain (see more in Section A.1.4 below), is as follows. If $\lim_{n\to\infty} p_{ij}^{(n)} = 0$ for every state $j \in S$, then the state i is null recurrent, whereas it is positive recurrent otherwise.

Returning to the case where S is finite, we find that every recurrent state is positive recurrent [17, p. 225]. Indeed, suppose that $N < \infty$ and $i \in S$ is a null recurrent state; that is, $\lim_{n\to\infty} p_{ij}^{(n)} = 0$ for every state $j \in S$. On the other hand, we have $\sum_{j\in S} p_{ij}^{(n)} = 1$ for all n, since $\left(p_{i1}^{(n)}, p_{i2}^{(n)}, \ldots, p_{iN}^{(n)}\right)$ is a probability vector. Using the fact that this summation is finite, it follows that

$$1 = \lim_{n\to\infty} (1) = \lim_{n\to\infty} \left(\sum_{j\in S} p_{ij}^{(n)} \right) = \sum_{j\in S} \left(\lim_{n\to\infty} p_{ij}^{(n)} \right) = \sum_{j\in S} (0) = 0,$$

a contradiction.

If a Markov chain with finite state space is irreducible, then all of its states must be (positive) recurrent. More generally, within a single communicating class, all states must either be transient or recurrent. To see this, suppose that two states $i \neq j$ belong to the same communicating class, so that there exists a positive integer n such $p_{ji}^{(n)} > 0$. It follows that if i is a recurrent state, then so is j, since any time the process reaches i there is a positive probability that it will reach j exactly n steps later. An analogous argument shows that if j is a recurrent state, then so is i.

In the reducible case, we have already seen an example involving both transient and recurrent states. Specifically, in Figure A.2b, state 1 is (positive) recurrent and states 2 - 4 are transient.

A.1.4 Long-term behavior and the stationary distribution

Of central importance in the study of Markov chains is the question of their long-term behavior. As we have previously remarked, a chain will not, in general, terminate at a single state (this only occurs if an absorbing state is reached), but it may still approach a sort of "steady-state" distribution over the state space.

Definition A.2. A vector $X_* = \left(X_*^{(j)}\right)_{j\in S}$ is called the *stationary distribution* of a Markov chain with transition matrix $P = (p_{ij})_{i,j\in S}$ on the finite state space S if it satisfies:

$$0 \le X_*^{(j)} \le 1, \quad j \in S, \tag{A.3}$$

$$\sum_{j\in S} X_*^{(j)} = 1, \tag{A.4}$$

$$X_*^{(i)} = \sum_{j\in S} p_{ij} X_*^{(j)}, \quad i \in S. \tag{A.5}$$

Whereas (A.3) and (A.4) simply assert that X_* is a probability vector, (A.5) is the key part of this definition. Rewritten in vector/matrix form, (A.5) is equivalent to $X_* = PX_*$, which implies that

$$P^2 X_* = P(PX_*) = PX_* = X_*,$$

and by a straightforward induction argument we conclude that

$$P^n X_* = X_*, \quad n = 0, 1, \ldots. \tag{A.6}$$

Given an initial distribution X_0, the next state of the chain is calculated as

$$X_1 = PX_0,$$

and the next state after that is

$$X_2 = PX_1 = P(PX_0) = P^2X_0.$$

In general, after n steps, the random variable representing the state of the process is

$$X_n = P^nX_0. \tag{A.7}$$

By (A.6) and (A.7), we see that if $X_0 = X_*$, then $X_n = X_*$ for all $n \geq 0$, so that the distribution of X_n is "stationary" with respect to time.

An alternate definition of the stationary distribution of a time-homogeneous Markov chain on a finite state space is

$$X_* = \lim_{n\to\infty} P^nX_0, \tag{A.8}$$

provided that the limit exists, where X_0 is any initial state and X_* satisfies (A.3) and (A.4) so as to be a probability vector. To see that (A.8) implies (A.5), we observe that

$$X_* = \lim_{n\to\infty} P^nX_0 = P\left(\lim_{n\to\infty} P^{n-1}X_0\right) = PX_*.$$

Thus, the stationary distribution, which may also be called the *invariant probability distribution* because of its invariance with respect to time, represents the long-term behavior of the process. In particular, each component, $X_*^{(i)}$, of the stationary distribution represents the probability of the chain will be at a particular state $i \in S$ in the long run.

An irreducible Markov chain has a stationary distribution if and only if all of its states are positive recurrent. Thus, an irreducible Markov chain with a finite state space is guaranteed to have a stationary distribution. If the chain is aperiodic as well as irreducible, then its stationary distribution is the limiting distribution defined by (A.8). This result is a special case of the Perron-Frobenius theorem [44, p. 667] and guarantees the existence of the PageRank vector as the stationary distribution of the Markov chain whose transition

probabilities depend on the hyperlink structure of the World Wide Web and whose finite state space is comprised of the individual pages within that web.

Appendix B

Martingales

This appendix is intended to complement Chapters 3 and 4 by providing an elementary introduction to the theory of discrete-time martingales. For more on martingales and stochastic processes in general, we refer again to the text of Lawler [40], as well as that of Bass [8].

Let $(\Omega, \mathcal{F}, \mathbb{P})$ be a probability space. A sequence of σ-algebras $\mathbb{F} = \{\mathcal{F}_0, \mathcal{F}_1, \dots\}$ is a *filtration* if $\mathcal{F}_n \subseteq \mathcal{F}_{n+1} \subseteq \mathcal{F}$ for n = 0, 1, ... If $\mathbb{F} = \{\mathcal{F}_0, \mathcal{F}_1, \dots\}$ is a filtration and $X = \{X_0, X_1, \dots\}$ is a discrete-time stochastic process, then X is said to be *adapted* to $\mathbb{F}$ if X_n is $\mathcal{F}_n$-measurable for $n = 0, 1, \dots$. If $\mathcal{F}_n = \sigma(X_0, \dots, X_n)$ for $n = 0, 1, \dots$, then $\mathbb{F}$ is called the *natural filtration* for X. By definition, every process is adapted to its natural filtration.

A *martingale* is a stochastic process that models a fair game, in that knowledge of past events never helps to predict expected future outcomes. In particular, a martingale is a sequence of random variables with the property that, at a given time, the expectation of the next value in the sequence, conditioned on the knowledge of all previously-observed values, is simply equal to the present observed state. Formally, we have the following definition.

Definition B.1. A sequence of random variables $X = \{X_0, X_1, \dots\}$ is a (discrete-time) *martingale* with respect to the filtration $\mathbb{F} = \{\mathcal{F}_0, \mathcal{F}_1, \dots\}$ if and only if X is adapted to $\mathbb{F}$

and satisfies:

$$\mathbb{E}\left[|X_n|\right] < \infty,\ n = 0, 1, \ldots \tag{B.1}$$

$$\mathbb{E}\left[X_{n+1} \mid \mathcal{F}_n\right] = X_n,\ n = 0, 1, \ldots. \tag{B.2}$$

We shall exclusively consider discrete-time martingales, although the definition can be extended to continuous-time. By the linearity property of expectation, the martingale property (B.2) can be rewritten as

$$\mathbb{E}\left[X_{n+1} - X_n \mid \mathcal{F}_n\right] = 0,\ n = 0, 1, \ldots.$$

Taking the expectation of both sides of (B.2) yields $\mathbb{E}\left[X_{n+1}\right] = \mathbb{E}\left[X_n\right]$ and it follows by induction that

$$\mathbb{E}\left[X_n\right] = \mathbb{E}\left[X_0\right],\ n = 0, 1, \ldots.$$

As another consequence of (B.2), we have the property that

$$\mathbb{E}\left[X_n \mid \mathcal{F}_m\right] = X_m,\ n = 0, 1, \ldots,$$

for every nonnegative integer $m \leq n$. To see this, we let $k \geq 0$ be such that $m = n - k$ and use the fact that $\{\mathcal{F}_0, \mathcal{F}_1, \ldots\}$ is an nondecreasing sequence of σ-algebras:

$$\begin{aligned}
\mathbb{E}\left[X_n \mid \mathcal{F}_m\right] &= \mathbb{E}\left[\mathbb{E}\left[X_n \mid \mathcal{F}_{n-1}\right] \mid \mathcal{F}_m\right] \\
&= \mathbb{E}\left[X_{n-1} \mid \mathcal{F}_m\right] \\
&\vdots \\
&= \mathbb{E}\left[\mathbb{E}\left[X_{n-k+2} \mid \mathcal{F}_{n-k+1}\right] \mid \mathcal{F}_m\right] \\
&= \mathbb{E}\left[X_{n-k+1} \mid \mathcal{F}_m\right] \\
&= X_m.
\end{aligned}$$

Since the notion of martingales comes from fair game betting strategies, we consider the following simple example.

Example B.1 (Gambler's Fortune)**.** Let X_n denote a gambler's fortune at time $n = 1, 2, \dots$. A coin is tossed and the gambler wins \$1 if the result is heads and loses \$1 if it is tails. In this scenario, the gambler's expected fortune after the next coin flip, given the history of all prior flips, is simply equal to their present fortune. Hence, the sequence $\{X_1, X_2, \dots\}$ is a martingale.

We conclude by stating a version of the strong law of large numbers for martingales. This is used in the proofs of Theorems 3.2.2 and 3.3.1, as well as in establishing Lemma 4.4.3.

Lemma B.0.1 (Martingale SLLN)**.** *Let $\mathbb{F} = \{\mathcal{F}_0, \mathcal{F}_1, \dots\}$ be a filtration and suppose that $\{Y_0, Y_1, \dots\}$ is a sequence of random variables with finite expectation such that Y_k is $\mathcal{F}_{k+1}$-measurable for $k = 0, 1, \dots$. Moreover, assume that $\mathbb{E}[Y_k \mid \mathcal{F}_k] = 0$, $k = 0, 1, \dots$, so that the sequence $\{M_1, M_2, \dots\}$ defined by*

$$M_k = \sum_{j=0}^{k-1} Y_k, \;\; k = 1, 2, \dots.$$

is a martingale with respect to $\mathbb{F}$. Then, if

$$\sum_{k=1}^{\infty} k^{-2}\, \mathbb{E} Y_k^2 < \infty,$$

we have

$$\lim_{k \to \infty} k^{-1} M_k = 0, \qquad a.s.$$

Appendix C

MATLAB Code

The following MATLAB scripts and functions may be used to reproduce the numerical simulations presented throughout this document. A brief description of each piece of code is provided in its header comment. The organization is as follows.

- Appendix C.1: basic PageRank computation.
  - C.1.1: script for computing PageRank via the power iteration.
  - C.1.2: function for generating a random web with no dangling nodes.
- Appendix C.2: estimation of an unknown parameter.
  - C.2.1: script for simulating parameter estimation procedures.
  - C.2.2: function for centralized least squares estimation of the parameter.
  - C.2.3: function for distributed least squares estimation of the parameter.
- Appendix C.3: optimal adaptive control procedures.
  - C.3.1: script for simulating the effect of various control policies.
  - C.3.2: function for minimum variance adaptive control.
  - C.3.3: function for adaptive tracking.
  - C.3.4: function for adaptive control under quadratic cost criterion.

C.1 PageRank Computation

C.1.1 Power Iteration Example

```
% ------------------------------------------------------------------------ %
% Script performs the power iteration on a simulated web hyperlink       %
% matrix in order to compute the PageRank vector.                         %
%                                                                         %
% Required file(s): websim.m                                              %
%                                                                         %
% Written by Cody E. Clifton, 2015-04-08.                                 %
% ------------------------------------------------------------------------ %

clear
n = 200; % size of the web
N = 20; % maximum number of iterations of the power method
t = .15; % teleportation parameter

% randomly generate hyperlink matrix
[H_vals,r_index,c_index,num_in] = websim(n);
H = sparse(r_index,c_index,H_vals,n,n);

% initialize matrix of PR estimates
X = zeros(n,N+1);
uniform = ones(n,1)/n;
X(:,1) = uniform;

% compute sequence of PR estimates
for iter = 2:N+1
    X(:,iter) = (1-t)*(H*X(:,iter-1)) + t*uniform;
    change = norm(X(:,iter)-X(:,iter-1));
end

```

```
% plot sequence of PR estimates
figure
hold off
for i=1:n
    plot(linspace(0,N,N+1),X(i,:),'LineWidth',3)
    hold on
    set(gca,'FontSize',20,'FontWeight','bold','XTick',0:N/4:N)
    xlabel('iterations','FontSize',28,'FontWeight','bold')
    ylabel('PageRank iterates','FontSize',28,'FontWeight','bold')
end % for i

% plot PR against page indices
figure
plot(linspace(1,n,n),X(:,iter),'o','MarkerSize',12,'MarkerFaceColor','k');
set(gca,'FontSize',20,'FontWeight','bold','XTick',0:n/5:n)
xlabel('page #','FontSize',28,'FontWeight','bold')
ylabel('PageRank value','FontSize',28,'FontWeight','bold')

% plot PR against number of in-links
figure
plot(num_in,X(:,iter),'o','MarkerSize',12,'MarkerFaceColor','k');
set(gca,'FontSize',20,'FontWeight','bold')
xlabel('# in-links','FontSize',28,'FontWeight','bold')
ylabel('PageRank value','FontSize',28,'FontWeight','bold')

% web graph sparsity diagram
figure
spy(H)
set(gca,'FontSize',20,'FontWeight','bold','XTick',0:n/5:n,'YTick',0:n/5:n)
xlabel('page #','FontSize',28,'FontWeight','bold')
ylabel('page #','FontSize',28,'FontWeight','bold')

% circular web graph diagram
```

```
figure
coords = [cos(2*pi*(1:n)/n); sin(2*pi*(1:n)/n)]';
gplot(H,coords,'-*')
for i = 1:n
    if mod(i,(n/10)) == 0
        text(1.1*coords(i,1)-0.08,1.1*coords(i,2),...
             sprintf('%d',i),'FontSize',28,'FontWeight','bold')
    end % if
end % for i
axis square off
```

C.1.2 Random Web Generation Function

```
function [H_vals,r_index,c_index,num_in] = websim(n)
% ------------------------------------------------------------------- %
% Function generates a simulated web hyperlink matrix according to a  %
% power law distribution.                                             %
%                                                                     %
% Input:    n           size of the web.                              %
%                                                                     %
% Output:   H_vals      nonzero elements of hyperlink matrix          %
%           r_index     row indices corresponding to H_vals           %
%           c_index     column indices corresponding to H_vals        %
%                                                                     %
% Written by Cody E. Clifton, 2015-04-08.                             %
% ------------------------------------------------------------------- %

% generate Pareto (power law) distribution for determining # in-links
dist_vec = (1:n-1).^(2.1); % (larger power => greater link sparsity)
dist_vec = 1./dist_vec;
dist_vec = dist_vec/sum(dist_vec);
cumulative_vec = cumsum(dist_vec);
```

```
20
21 % randomly generate each page's number of in-links
22 num_in = zeros(n,1);
23 for i=1:n
24     r = rand;
25     for j = 1:n-1
26         if r <= cumulative_vec(j)
27             break
28         end % if
29     end % for j
30     num_in(i) = j;
31 end % for i
32
33 total_in = sum(num_in); % total number of links
34
35 % initialize vectors to hold indices of nonzero elements of H
36 r_index = zeros(total_in,1);
37 c_index = zeros(total_in,1);
38
39 % randomly generate where in-links come from
40 c = 0; % temp link counter
41 for i = 1:n
42     j = num_in(i);
43     r_index(c+1:c+j) = i*ones(j,1);
44     c_index_i = zeros(j,1);
45     for k = 1:j
46         while true
47             r = fix(n*rand)+1;
48             if ~((r == i) || any(c_index_i == r)) % avoid self-links & ...
                   repeats
49                 c_index_i(k) = r;
50                 break
51             end % if
```

```
52          end % while
53      end % for k
54      c_index(c+1:c+j) = c_index_i;
55      c = c + j;
56  end % for i
57
58  % "back-link" from dangling nodes
59  for i = 1:n
60      if ~any(c_index == i)
61          I = find(r_index == i);
62          r_index = cat(1,r_index,c_index(I));
63          c_index = cat(1,c_index,i*ones(length(I),1));
64      end % if
65  end % for i
66
67  % total number of outlinks from each page
68  num_out = full(sum(sparse(r_index,c_index,1,n,n)));
69
70  H_vals = 1./num_out(c_index);
71
72  end % function
```

C.2 Parameter Estimation

C.2.1 Estimation Example

```
1  % ------------------------------------------------------------------- %
2  % Script performs an example of least squares estimation of an unknown  %
3  % parameter, alpha, within a stochastic system model for PageRank.      %
4  %                                                                       %
5  % Required files: websim.m & one of lse.m or dist_lse.m                 %
6  %                                                                       %
```

```
7  % Written by Cody E. Clifton, 2015-04-08.                              %
8  % ------------------------------------------------------------------- %
9
10 clear
11 n = 200; % size of the web
12 N = 20; % max number of iterations of power method / LS estimation
13 t = .15; % teleportation parameter
14 alpha0 = 1/2; % true value of unknown parameter
15 var = 1e-5; %random noise variance
16
17 q = length(alpha0);
18
19 % randomly generated distributed link matrices
20 [H_vals,r_index,c_index,num_in] = websim(n);
21 r1 = rand(1,length(H_vals));
22 r2 = rand(q,length(H_vals));
23 r2 = bsxfun(@rdivide,r2,sum(r2,1));
24 r2 = bsxfun(@times,r2,r1);
25 F0_vals = H_vals.*(1-r1);
26 F_vals = zeros(q,length(H_vals));
27 for i = 1:q
28     F_vals(i,:) = (H_vals.*r2(i,:))/alpha0(i);
29 end % for i
30
31 % ------------------------------------------------------------------- %
32 % Centralized LSE (requires lse.m). Uncomment below to apply.         %
33 % ------------------------------------------------------------------- %
34 % H_vals_alpha = @(alpha) F0_vals;
35 % for i = 1:q
36 %     H_vals_alpha = @(alpha) H_vals_alpha(alpha) + alpha(i).*F_vals(i,:);
37 % end % for i
38 % H = @(alpha) sparse(r_index,c_index,H_vals_alpha(alpha),n,n);
39 % [X,alpha_hat] = lse(n,N,t,alpha0,H,var);
```

```
% ------------------------------------------------------------------------ %

% ------------------------------------------------------------------------ %
% Distributed LSE (requires dist_lse.m). Uncomment below to apply.         %
% ------------------------------------------------------------------------ %
% F0 = sparse(r_index,c_index,F0_vals,n,n);
% F1 = sparse(r_index,c_index,F_vals(1,:),n,n);
% F_dist = zeros(n,n,n,2);
% for j = 1:n
%     for k = 1:n
%         for l = 1:n
%             if (j==k) || (j==l);
%                 F_dist(k,l,j,1) = F0(k,l);
%                 F_dist(k,l,j,2) = F1(k,l);
%             elseif (k==l) && (j~=k);
%                 F_dist(k,l,j,1) = 1 - F0(j,l);
%                 F_dist(k,l,j,2) = - F1(j,l);
%             end % if/else
%         end % for l
%     end % for k
% end % for j
% [X,alpha_hat] = dist_lse(n,N,t,alpha0,F_dist,var);
% ------------------------------------------------------------------------ %

s = linspace(0,N,N+1);

% plot sequence of PR estimates
figure
hold off
for i=1:n
    plot(s,X(i,:),'-*','LineWidth',3)
    hold on
    set(gca,'FontSize',20,'FontWeight','bold')
```

```
73      xlabel('iterations','FontSize',28,'FontWeight','bold')
74      ylabel('PageRank iterates','FontSize',28,'FontWeight','bold')
75  end % for i
76
77  % plot sequence of LS parameter estimates
78  figure
79  plot(s,alpha_hat,'-*',s,bsxfun(@times,alpha0,ones(N+1,q)),'LineWidth',3)
80  set(gca,'FontSize',20,'FontWeight','bold')
81  xlabel('iterations','FontSize',28,'FontWeight','bold')
82  ylabel('parameter estimates','FontSize',28,'FontWeight','bold')
```

C.2.2 Centralized Least Squares Estimation Function

```
1  function [X,alpha_hat] = lse(n,N,t,alpha0,H,var)
2  % ------------------------------------------------------------------- %
3  % Function performs centralized least squares estimation of an unknown  %
4  % parameter, alpha, within a stochastic system model for PageRank.      %
5  %                                                                       %
6  % Input:    n           size of the web                                 %
7  %           N           max number of iterations of power method        %
8  %           t           teleportation parameter                         %
9  %           alpha0      true value of unknown parameter                 %
10 %           H           hyperlink matrix as function of alpha           %
11 %           var         variance of random noise in the system          %
12 %                                                                       %
13 % Output:   X           sequence of PageRank estimates                  %
14 %           alpha_hat   sequence of least squares estimates of alpha    %
15 %                                                                       %
16 % Written by Cody E. Clifton, 2015-04-08.                               %
17 % ------------------------------------------------------------------- %
18
19 % construct "true" hyperlink matrix (assume knowledge of alpha0)
```

```
20 H0 = H(alpha0);
21
22 % initialize matrix of PR estimates
23 X = zeros(n,N+1);
24 uniform = ones(n,1)/n;
25 X(:,1) = uniform;
26
27 % initialize matrix of LS parameter estimates
28 alpha_hat = zeros(N+1,length(alpha0));
29
30 R = speye(n); % controller-selected matrix for modifying LS functional
31
32 % generate random noise
33 xi = randn(n,N+1)*sqrt(var);
34
35 % initialize LS functional
36 L = @(alpha) 0;
37
38 % compute sequences of PR estimates & LS parameter estimates
39 for i=2:N+1
40     X(:,i) = (1-t)*H0*X(:,i-1) + t*uniform + xi(:,i);
41     L = @(alpha) L(alpha) + ...
42         transpose(X(:,i) - (1-t)*H(alpha)*X(:,i-1) - t*uniform)...
43         *R*(X(:,i) - (1-t)*H(alpha)*X(:,i-1) - t*uniform);
44     alpha_hat(i,:) = fminsearch(L,zeros(length(alpha0),1));
45 end % for i
46
47 end % function
```

C.2.3 Distributed Least Squares Estimation Function

```
1 function [X,alpha_hat] = dist_lse(n,N,t,alpha0,F_dist,var)
```

```
% ------------------------------------------------------------------- %
% Function performs distributed least squares estimation of an unknown  %
% parameter, alpha, within a stochastic system model for PageRank.      %
%                                                                       %
% Input:    n           size of the web                                 %
%           N           max number of iterations of power method        %
%           t           teleportation parameter                         %
%           alpha0      true value of unknown parameter                 %
%           F_dist      distributed (known) matrices                    %
%           var         variance of random noise in the system          %
%                                                                       %
% Output:   X           sequence of PageRank estimates                  %
%           alpha_hat   sequence of least squares estimates of alpha    %
%                                                                       %
% Written by Cody E. Clifton, 2015-04-08.                               %
% ------------------------------------------------------------------- %

t_hat = (2*t)/(n - t*(n-2)); % distributed teleportation parameter

% construct "true" distr. hyperlink matrices (assume knowledge of alpha0)
H0_dist = F_dist(:,:,:,1) + alpha0*F_dist(:,:,:,2);

% initialize matrix of PR estimates
X = zeros(n,N+1);
uniform = ones(n,1)/n;
X(:,1) = uniform;

% initialize matrix of distributed LS parameter estimates
alpha_hat = zeros(N+1,1);

R = speye(n); % controller-selected matrix for modifying LS functional

% generate random noise
```

```
xi = randn(n,N+1)*sqrt(var);

% initialize distributed LS functional
L = @(alpha) 0;

% compute sequences of distributed PR estimates & LS parameter estimates
for i=2:N+1
    theta = unidrnd(n);
    X(:,i) = (1-t_hat)*H0_dist(:,:,theta)*X(:,i-1) ...
                          + t_hat*uniform + xi(:,i);
    H_theta = @(alpha) F_dist(:,:,theta,1) + alpha*F_dist(:,:,theta,2);
    L = @(alpha) L(alpha) + transpose(X(:,i) ...
                    - (1-t_hat)*H_theta(alpha)*X(:,i-1) - t_hat*uniform)...
        *R*(X(:,i) - (1-t_hat)*H_theta(alpha)*X(:,i-1) - t_hat*uniform);
    alpha_hat(i,:) = fminsearch(L,zeros(length(alpha0),1));
end % for i

end % function
```

C.3 Adaptive Control

C.3.1 Optimal Adaptive Control Example

```
% ------------------------------------------------------------------- %
% Script applies one of three control procedures to a stochastic system %
% model for PageRank (PR) that depends on an unknown parameter alpha.   %
% Four sequences are plotted over time: PR estimates, LS estimates of   %
% alpha, average control cost values, and the normed diffences between  %
% the optimal stationary control and the approximated optimal control.  %
%                                                                       %
% Required files: websim.m & one of MVctrl.m, tracking.m, or LQctrl.m   %
%                                                                       %
```

```
% Written by Cody E. Clifton, 2015-04-10.                                %
% ------------------------------------------------------------------- %

clear
n = 10; % size of the web
N = 100; % max number of iterations of power method / LS estimation
t = .15; % teleportation parameter
alpha0 = 1/2; % true value of unknown parameter
var = 1e-5; % random noise variance

% randomly generated distributed link matrices
[H_vals,r_index,c_index,num_in] = websim(n);
r = rand(1,length(H_vals));
F0_vals = H_vals.*(1.-r);
F1_vals = (H_vals.*r)/alpha0;
F0 = sparse(r_index,c_index,F0_vals,n,n);
F1 = sparse(r_index,c_index,F1_vals,n,n);

% define general hyperlink matrix as a function of alpha
H = @(alpha) sparse(r_index,c_index,F0_vals+alpha.*F1_vals,n,n);

% ------------------------------------------------------------------- %
% MV control (requires MVctrl.m). Uncomment below to apply.           %
% ------------------------------------------------------------------- %
% [X,alpha_hat,K,Knormdiff,C,optC] = MVctrl(n,N,t,alpha0,H,var);
% Y = zeros(n,1);
% ------------------------------------------------------------------- %

% ------------------------------------------------------------------- %
% Adaptive tracking (requires tracking.m). Uncomment below to apply.  %
% ------------------------------------------------------------------- %
% [X,Y,alpha_hat,K,Knormdiff,C,optC] = tracking(n,N,t,alpha0,H,var);
% ------------------------------------------------------------------- %
```

```
43
44 % ------------------------------------------------------------------------ %
45 % Quadratic cost control (requires LQctrl.m). Uncomment below to apply. %
46 % ------------------------------------------------------------------------ %
47 % [X,alpha_hat,K,Knormdiff,C,optC] = LQctrl(n,N,t,alpha0,H,var);
48 % Y = zeros(n,1);
49 % ------------------------------------------------------------------------ %
50
51 s = linspace(0,N,N+1);
52
53 % plot sequence of PR estimates
54 figure
55 hold off
56 plot(s,Y*ones(1,N+1),'LineWidth',3)
57 hold on
58 for i=1:n
59     plot(s,X(i,:),'-*','LineWidth',3)
60     hold on
61     set(gca,'FontSize',20,'FontWeight','bold','XTick',0:N/4:N)
62     xlabel('iterations','FontSize',28,'FontWeight','bold')
63     ylabel('PageRank iterates','FontSize',28,'FontWeight','bold')
64 end % for i
65
66 % plot sequence of LS parameter estimates
67 figure
68 plot(s,alpha_hat,'-*',s,alpha0*ones(N+1,1),'LineWidth',3)
69 set(gca,'FontSize',20,'FontWeight','bold','XTick',0:N/4:N)
70 xlabel('iterations','FontSize',28,'FontWeight','bold')
71 ylabel('parameter estimates','FontSize',28,'FontWeight','bold')
72
73 avgC = C./transpose(linspace(1,N+1,N+1));
74
75 % plot sequence of average cost values
```

```
figure
plot(s,avgC,'-*',s,optC,'LineWidth',3)
set(gca,'FontSize',20,'FontWeight','bold','XTick',0:N/4:N)
xlabel('iterations','FontSize',28,'FontWeight','bold')
ylabel('average cost','FontSize',28,'FontWeight','bold')

% plot sequence of norm(K(alpha_hat(i)) - K(alpha0))
figure
plot(linspace(0,N-1,N),Knormdiff,'LineWidth',3)
set(gca,'FontSize',20,'FontWeight','bold','XTick',0:N/4:N)
xlabel('iterations','FontSize',28,'FontWeight','bold')
ylabel('norm(K-K0)','FontSize',28,'FontWeight','bold')
```

C.3.2 Minimum Variance Control Function

```
function [X,alpha_hat,K,Knormdiff,C,optC] = MVctrl(n,N,t,alpha0,H,var)
% ------------------------------------------------------------------------ %
% Function applies minimum variance adaptive control to a stochastic      %
% system model for PageRank that depends on an unknown parameter alpha.   %
%                                                                          %
% Input:    n           size of the web                                    %
%           N           max number of iterations of power method           %
%           t           teleportation parameter                            %
%           alpha0      true value of unknown parameter                    %
%           H           hyperlink matrix as function of alpha              %
%           var         variance of random noise in the system             %
%                                                                          %
% Output:   X           sequence of PageRank estimates                     %
%           alpha_hat   sequence of least squares estimates of alpha       %
%           K           control policy                                     %
%           Knormdiff   sequence of normed differences between optimal     %
%                       stationary control and approx. optimal control     %
```

```
% C sequence of control cost values %
% optC cost of optimal stationary control %
% %
% Written by Cody E. Clifton, 2015-04-09. %
% ---------------------------------------------------------------------- %

% construct "true" hyperlink matrix (assume knowledge of alpha0)
H0 = H(alpha0);

uniform = ones(n,1)/n;

% initialize the matrix of PR estimates
X = zeros(n,N+1); X(:,1) = uniform;

% initialize matrix of LS parameter estimates
alpha_hat = zeros(N+1,1);

R = speye(n); % controller-selected matrix for modifying LS functional

% generate random full-rank (square) matrix
S = expm(randn(n));

% compute a left-inverse of S
SLinv = sparse(((S.')*S)\(S.'));

% define control matrix as function of alpha
K = @(alpha) -SLinv*((1-t)*H(alpha) + (t/n)*ones(n));

% initialize vector for norm(K(alpha_hat(i)) - K(alpha0))
Knormdiff = zeros(N,1);

% initialize vector of cost criterion iterates
C = zeros(N+1,1); C(1) = norm(X(:,1))^2;
```

```
51
52 % optimal cost
53 optC = n*var*ones(N+1,1);
54
55 % generate random noise
56 xi =  randn(n,N+1)*sqrt(var);
57
58 % initialize LS functional
59 L = @(alpha) 0;
60
61 % compute sequences: X, alpha_hat, C, Knormdiff
62 for i=2:N+1
63     X(:,i) = ((1-t)*H0 + (t/n)*ones(n)...
64                   + S*K(alpha_hat(i-1)))*X(:,i-1) + xi(:,i);
65     C(i) = C(i-1) + norm(X(:,i))^2;
66     L = @(alpha) L(alpha) + ...
67         transpose(X(:,i) - (1-t)*H(alpha)*X(:,i-1)...
68                       - t*uniform - S*K(alpha_hat(i-1))*X(:,i-1))...
69                   *R*(X(:,i) - (1-t)*H(alpha)*X(:,i-1)...
70                       - t*uniform - S*K(alpha_hat(i-1))*X(:,i-1));
71     Knormdiff(i-1) = norm(K(alpha_hat(i-1)) - K(alpha0));
72     alpha_hat(i) = fminbnd(L,0,1);
73 end % for i
74
75 end % function
```

C.3.3 Adaptive Tracking Function

```
1 function [X,Y,alpha_hat,K,Knormdiff,C,optC] = tracking(n,N,t,alpha0,H,var)
2 % ------------------------------------------------------------------------ %
3 % Function applies an adaptive control, constructed under a quadratic   %
4 % cost criterion, to a stochastic system model for PageRank (PR) that   %
```

```
% depends on an unknown parameter alpha.                                  %
%                                                                         %
% Input:    n           size of the web                                   %
%           N           max number of iterations of power method          %
%           t           teleportation parameter                           %
%           alpha0      true value of unknown parameter                   %
%           H           hyperlink matrix as function of alpha             %
%           var         variance of random noise in the system            %
%                                                                         %
% Output:   X           sequence of PageRank estimates                    %
%           Y           reference signal to be tracked                    %
%           alpha_hat   sequence of least squares estimates of alpha      %
%           K           control policy                                    %
%           Knormdiff   sequence of normed differences between optimal    %
%                       stationary control and approx. optimal control    %
%           C           sequence of control cost values                   %
%           optC        cost of optimal stationary control                %
%                                                                         %
% Written by Cody E. Clifton, 2015-04-13.                                 %
% ----------------------------------------------------------------------- %

% construct "true" hyperlink matrix (assume knowledge of alpha0)
H0 = H(alpha0);

uniform = ones(n,1)/n;

% initialize the matrix of PageRank vector estimates
X = zeros(n,N+1); X(:,1) = uniform;

% randomly generate reference signal
Y = rand(n,1); Y = Y./sum(Y);

% initialize vector of LS estimates
```

```
38 alpha_hat = zeros(N+1,1);
39
40 R = speye(n); % controller-selected matrix for modifying LS functional
41
42 S = eye(n);
43
44 % define control term as function of alpha
45 K = @(alpha) Y*ones(1,n) - ((1-t)*H(alpha) + (t/n)*ones(n));
46
47 % initialize vector for norm(K(alpha_hat(i)) - K(alpha0))
48 Knormdiff = zeros(N,1);
49
50 % initialize vector of cost criterion iterates
51 C = zeros(N+1,1); C(1) = norm(X(:,1) - Y)^2;
52
53 % optimal cost
54 optC = n*var*ones(N+1,1);
55
56 % generate random noise
57 xi = randn(n,N+1)*sqrt(var);
58
59 % initialize LS functional
60 L = @(alpha) 0;
61
62 % compute sequences: X, alpha_hat, C, Knormdiff
63 for i=2:N+1
64     X(:,i) = (1-t)*H0*X(:,i-1) + t*uniform...
65                 + S*K(alpha_hat(i-1))*X(:,i-1) + xi(:,i);
66     X(:,i) = X(:,i)./sum(X(:,i),1);
67     C(i) = C(i-1) + norm(X(:,i) - Y)^2;
68     L = @(alpha) L(alpha) + ...
69         transpose(X(:,i) - (1-t)*H(alpha)*X(:,i-1)...
70                     - t*uniform - S*K(alpha_hat(i-1))*X(:,i-1))...
```

```
                    *R*(X(:,i) - (1-t)*H(alpha)*X(:,i-1)...
                      - t*uniform - S*K(alpha_hat(i-1))*X(:,i-1));
    Knormdiff(i-1) = norm(K(alpha_hat(i-1)) - K(alpha0));
    alpha_hat(i) = fminbnd(L,0,1);
end % for i

end % function
```

C.3.4 Quadratic Cost Control Function

```
function [X,alpha_hat,K,Knormdiff,C,optC] = LQctrl(n,N,t,alpha0,H,var)
% ------------------------------------------------------------------------ %
% Function applies an adaptive control, constructed under a quadratic     %
% cost criterion, to a stochastic system model for PageRank that           %
% depends on an unknown parameter alpha.                                   %
%                                                                          %
% Input:    n           size of the web                                    %
%           N           max number of iterations of power method           %
%           t           teleportation parameter                            %
%           alpha0      true value of unknown parameter                    %
%           H           hyperlink matrix as function of alpha              %
%           var         variance of random noise in the system             %
%                                                                          %
% Output:   X           sequence of PageRank estimates                     %
%           alpha_hat   sequence of least squares estimates of alpha       %
%           K           control policy                                     %
%           Knormdiff   sequence of normed differences between optimal     %
%                       stationary control and approx. optimal control     %
%           C           sequence of control cost values                    %
%           optC        cost of optimal stationary control                 %
%                                                                          %
% Written by Cody E. Clifton, 2015-04-09.                                  %
```

```
% ------------------------------------------------------------------------ %

% construct "true" hyperlink matrix (assume knowledge of alpha0)
H0 = H(alpha0);

uniform = ones(n,1)/n;

% initialize the matrix of PR estimates
X = zeros(n,N+1); X(:,1) = uniform;

% initialize matrix of LS parameter estimates
alpha_hat = zeros(N+1,1);

R = speye(n); % controller-selected matrix for modifying LS functional

% controller-selected matrices for modifying quadratic cost criterion
A = eye(n);
B = eye(n);

% initialize control matrix
K = -0.001*ones(n);

% generate random full-rank (square) matrix
S = expm(randn(n));
while max(abs(eig((1-t)*H(alpha0) + (t/n)*ones(n)+S*K))) >= 0.95
    S = expm(randn(n)); % ensure that Assumption 4.3 is satisfied
end

% compute optimal stationary control
for k = 1:10
    W = 0;
    for j = 1:1e4
        newW = W + transpose(((1-t)*H(alpha0)...
```

```
56                  + (t/n)*ones(n)+S*K)^j)*(A ...
57                  + transpose(K)*B*K)*(((1-t)...
58                  *H(alpha0) + (t/n)*ones(n)+S*K)^j);
59          if norm(newW-W) < 1e-8
60              W = newW;
61              break
62          end % if
63          W = newW;
64      end % for j
65      K = -(inv(B + transpose(S)*W*S))*transpose(S)...
66            *W*((1-t)*H(alpha0) + (t/n)*ones(n));
67  end % for k
68
69  trueK = K;
70
71  % initialize vector for norm(K(alpha_hat(i)) - K(alpha0))
72  Knormdiff = zeros(N,1);
73
74  % re-initialize control matrix
75  K = -0.001*ones(n);
76
77  % initialize vector of cost criterion iterates
78  C = zeros(N+1,1);
79  C(1) = transpose(X(:,1))*(A + transpose(K)*B*K)*X(:,1);
80
81  % optimal cost
82  optC = n*var*trace(W)*ones(N+1,1);
83
84  % generate random noise
85  xi = randn(n,N+1)*sqrt(var);
86
87  % initialize LS functional
88  L = @(alpha) 0;
```

```
89
90  % compute sequences: X, alpha_hat, C, Knormdiff
91  for i=2:N+1
92      X(:,i) = ((1-t)*H0 + (t/n)*ones(n) + S*K)*X(:,i-1) + xi(:,i);
93      C(i) = C(i-1) + transpose(X(:,i))*(A + transpose(K)*B*K)*X(:,i);
94      L = @(alpha) L(alpha) + ...
95          transpose(X(:,i) - ((1-t)*H(alpha) + (t/n)*ones(n) + ...
                S*K)*X(:,i-1))...
96                      *R*(X(:,i) - ((1-t)*H(alpha) + (t/n)*ones(n) + ...
                            S*K)*X(:,i-1));
97      alpha_hat(i) = fminbnd(L,0,1);
98      % compute approximated optimal control based on alpha_hat(i)
99      for k = 1:10
100         W = 0;
101         for j = 1:1e2
102             newW = W + transpose(((1-t)*H(alpha_hat(i))...
103                     + (t/n)*ones(n)+S*K)^j)*(A ...
104                     + transpose(K)*B*K)*(((1-t)...
105                     *H(alpha_hat(i)) + (t/n)*ones(n)+S*K)^j);
106             if norm(newW-W) < 1e-8
107                 W = newW;
108                 break
109             end % if
110             W = newW;
111         end % for j
112         K = -(inv(B + transpose(S)*W*S))*transpose(S)...
113               *W*((1-t)*H(alpha_hat(i)) + (t/n)*ones(n));
114     end % for k
115     Knormdiff(i-1) = norm(K - trueK);
116 end % for i
117
118 end % function
```